НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
Кафедра автоматизованих систем обробки інформації та управління

До захисту допущено:
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім'я, прізвище)
"___"_____2020 р.

Дипломний проєкт
на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Інформаційна система підтримки проведення творчих письмових конкурсів»

Виконав: студент *IV* курсу, групи *ІС-63*

Лопата Владислав Владиславович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник *доц., к.т.н., доц. Тєлишева Тамара Олексіївна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Консультант з графічної документації *ст. вик. Проскура Світлана Леонідівна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент *доц., к.т.н., доц Чумаченко Олена Іллівна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації та управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки та інформаційні технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім’я, прізвище)
“___”_____2020 р.

ЗАВДАННЯ
на дипломний проєкт студенту

Лопаті Владиславу Владиславовичу
(прізвище, ім’я, по батькові)

1. Тема проєкту «Інформаційна система підтримки проведення творчих письмових конкурсів»
керівник проєкту *Тєлишева Тамара Олексіївна, к.т.н., доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “*7*”*травня* 2020 р. №1081-с
2. Термін подання студентом проєкту “*01*”*червня* 2020 року
3. Вихідні дані до проєкту
Технічне завдання

4. Зміст пояснювальної записки
1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд ринку програмних продуктів, постановка задачі
2. Інформаційне забезпечення: вхідні дані, вихідні дані, опис структури бази даних
3. Математичне забезпечення: змістовна та математична постановки задачі, обґрунтування та опис методу розв’язання
4. Програмне та технічне забезпечення: засоби розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення, побудова звітів

5. Технологічний розділ: керівництво користувача, методика випробувань програмного продукту

5. Перелік графічного матеріалу

1. Схема структурна варіантів використань

2. Схема структурна послідовності

3. Схема структурна послідовності

4. Схема структурно-функціональна процесу проведення конкурсу в нотаціях IDEF0

5. Схема структурна класів програмного забезпечення

6. Схема структурна класів програмного забезпечення

7. Схема алгоритму

8. Креслення вигляду екранних форм

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання *«13» квітня 2020 року*

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	*Вивчення рекомендованої літератури*	*17.04.2020*	
2.	*Аналіз існуючих методів розв'язання задачі*	*20.04.2020*	
3.	*Постановка та формалізація задачі*	*23.04.2020*	
4.	*Розробка інформаційного забезпечення*	*30.04.2020*	
5.	*Алгоритмізація задачі*	*05.05.2020*	
6.	*Обґрунтування використовуваних технічних засобів*	*10.05.2020*	
7.	*Розробка програмного забезпечення*	*20.05.2020*	
8.	*Налагодження програми*	*25.05.2020*	
9.	*Виконання графічних документів*	*27.05.2020*	
10.	*Оформлення пояснювальної записки*	*28.05.2020*	
11.	*Подання ДП на попередній захист*	*15.05.2020*	
12.	*Подання ДП на основний захист*	*01.06.2020*	
13.	*Подання ДП рецензенту*	*02.06.2020*	

Студент Владислав ЛОПАТА

Керівник Тамара ТЄЛИШЕВА

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ДП 6314.00.000 ПЗ	Пояснювальна записка	68	
3	А4	ДП 6314.01.000 ТЗ	Технічне завдання	10	
4	А3	ДП 6314.02.000 ССВ	Схема структурна варіантів використань	1	
5	А3	ДП 6314.03.000 ССП	Схема структурна послідовності без	1	
			системи		
6	А3	ДП 6314.04.000 ССП	Схема структурна послідовності	1	
7	А3	ДП 6314.05.000	Схема структурно-функціональна процесу	1	
			проведення конкурсу в нотаціях IDEF0		
8	А3	ДП 6314.06.000 ССК	Схема структурна класів програмного	1	
			забезпечення		
9	А3	ДП 6314.07.000 ССК	Схема структурна класів програмного	1	
			забезпечення		
10	А3	ДП 6314.08.000 СА	Схема алгоритму	1	
11	А3	ДП 6314.09.000 КЕ	Креслення вигляду екранних форм	1	

Зм.	Арк.	ПІБ	Підп.	Дата	ДП 6314 00.000.00		
Розробн.		Лопата В. В.			Відомість дипломного проєкту	Літ.	Лист 1 / Листів 1
Керівн.		Тєлишева Т. О.					
Консульт.						КПІ ім. Ігоря Сікорського Каф. АСОІУ Гр. ІС-63	
Н/контр.		Проскура С.Л,					
В.о.зав.каф.		Павлов О.А.					

Пояснювальна записка
до дипломного проєкту

на тему: Інформаційна система підтримки проведення творчих письмових конкурсів

Київ – 2020 року

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проєкту складається з шести розділів, містить 15 рисунків, 11 таблиць, 8 додатків, 9 джерел.

Дипломний проєкт присвячений розробці інформаційної системи підтримки проведення творчих письмових конкурсів (ІСТПК).

Ціль проєкта: покращити організацію, процеси проведення і оцінювання конкурсних творчих робіт за рахунок автоматизації видачі завдань та їх оцінювання.

У першому розділі приведено опис предметного середовища, огляд наявних аналогів, інформаційного забезпечення і постановка задач.

Розділ математичного забезпечення присвячений побудові рейтингу учасників відбіркового етапу конкурсу на основі статистичної інформації про проходження учасниками тестів.

Програмне забезпечення, що представляє з себе клієнт-серверний за стосунок і вирішує поставлені задачі, описане у відповідному розділі.

У технологічному розділі увага присвячена тестуванню та керівництву користувача.

КОНКУРС, УЧАСНИК, ОРГАНІЗАТОР, ЖУРІ, РАНЖУВАННЯ, РЕЙТИНГ, ОЦІНЮВАННЯ

ABSTRACT

Structure and volume. The explanatory note of the diploma project consists of six sections, contains 15 figures, 11 tables, 8 appendixes, 9 sources.

The diploma project is devoted to the development of an information system to support creative writing competitions (ISSCWC).

The purpose of the project: to improve the organization, processes of conducting and evaluation of competitive creative works by automating the issuance of tasks and their evaluation.

In the section of information support the description of the subject environment and the review of available analogues and statement of a problem are resulted.

The section of mathematical support is devoted to the construction of the rating of the participants of the qualifying stage of the competition on the basis of statistical information about the participants' passing the tests.

The software, which is a client-server application, solves the tasks and is described in the relevant section.

In the technological section, attention is paid to testing and user guidance.

COMPETITION, PARTICIPANT, ORGANIZER, JURY, RANKING, RATING, EVALUATION

ЗМІСТ

ВСТУП

Творчі письмові конкурси користуються великою популярністю в школах та ВНЗ. Зокрема до такої різновидності конкурсів можна віднести олімпіади з української та іноземних мов, випускний екзамен «Зовнішнє незалежне оцінювання» (а саме ті ЗНО, які передбачають написання особистих висловлень) та шкільні конкурси з написання творів імені Тараса Григоровича Шевченка.

Актуальність теми проекту. В зв'язку з тим, що творчі письмові конкурси мають таку значну популярність, виникає необхідність в розробці інструментарій, що могли б надати змогу користувачам самостійно створювати конкурси, приймати в конкурсах участь а також надати можливість членам експертного журі виконувати процес суддівської оцінки конкурсних робіт. Нажаль, наразі програмних продуктів, що забезпечували б увесь необхідний функціонал для проведення конкурсів не існує. Через це виникає необхідність такий продукт створити.

Практичне значення одержаних результатів. Система підтримки проведення творчих письмових конкурсів корисна для організаторів творчих письмових конкурсів, оскільки вона надає можливість в автоматичному режимі розсилати завдання та формувати рейтинг учасників.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Творчі письмові конкурси можливо розподілити на ряд етапів.

Перший етап – відбірковий. У випадку, якщо на конкурс було зареєстровано багато учасників, і одразу перевірити усі їх роботи не є можливим, то необхідно спочатку провести відбір, наприклад, у вигляді тестового завдання з певної області.

Наступний етап – написання творчої роботи. Варіантів проведення такого етапу існує декілька.

Перший варіант – загальний конкурс. Він проходить в один тур і судді, оцінюючи роботи одразу формують фінальний рейтинг, з якого можна визначити переможців.

Другий варіант проведення основного етапу – турнірна система, де учасники поділяються на пари або групи і журі обирає кращого з групи, котрий перейде у наступний тур конкурсу.

Третій варіант – карусель, коли учасників розбивають на пари і журі обирає кращого, але при цьому гірший покидає конкурс лише після фіксованого числа поразок (двох та більше). Цей варіант в останній час набирає все більшу популярність в різних конкурсах.

На рисунках 1.1 та 1.2 показано декомпозицію середовища проведення творчих письмових конкурсів.

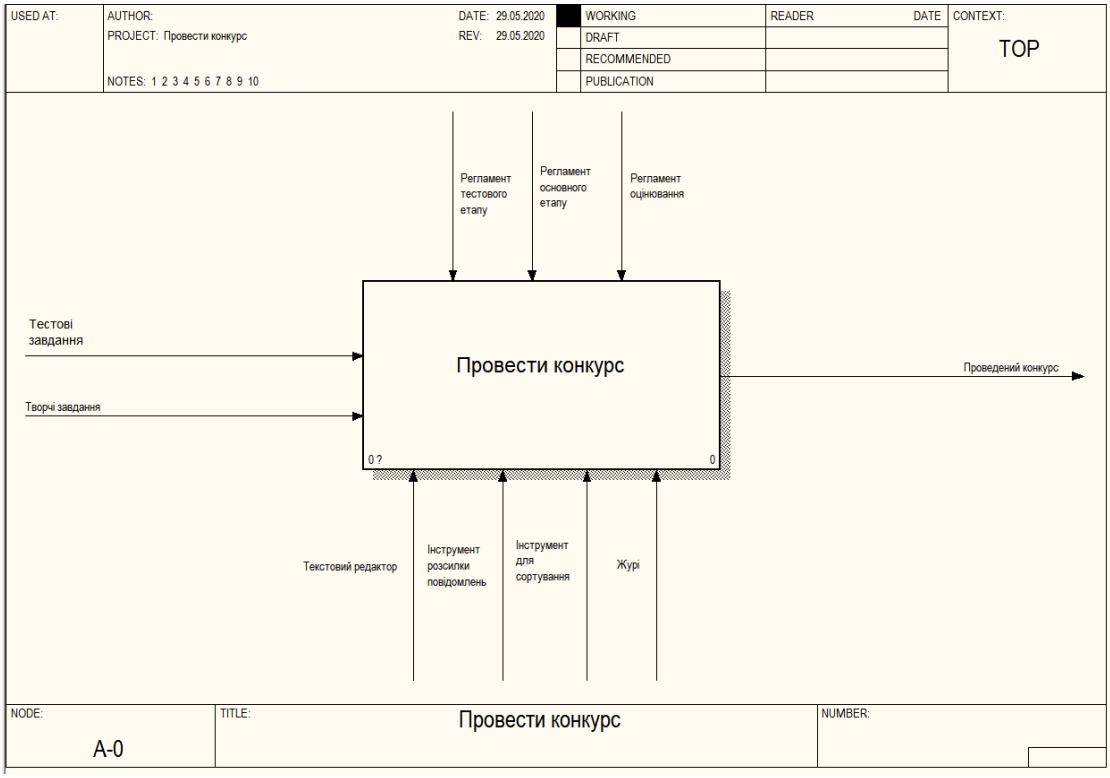


Рисунок 1.1 – Схема структурно-функціональна процесу проведення конкурсу в нотаціях IDEF0

Як показано на діаграмі на рисунку 1.1, для проведення конкурсу необхідні матеріали у вигляді тестових завдань та творчих завдань. Проведення конкурсу регулюється регламентами тестового та основного етапів, а також регламентом оцінювання робіт. Механізмами, що необхідні для проведення конкурсів слугують текстовий редактор, інструмент розсилки повідомлень, інструмент для сортування та журі. Вихідними даними проведення конкурсу є проведений конкурс.

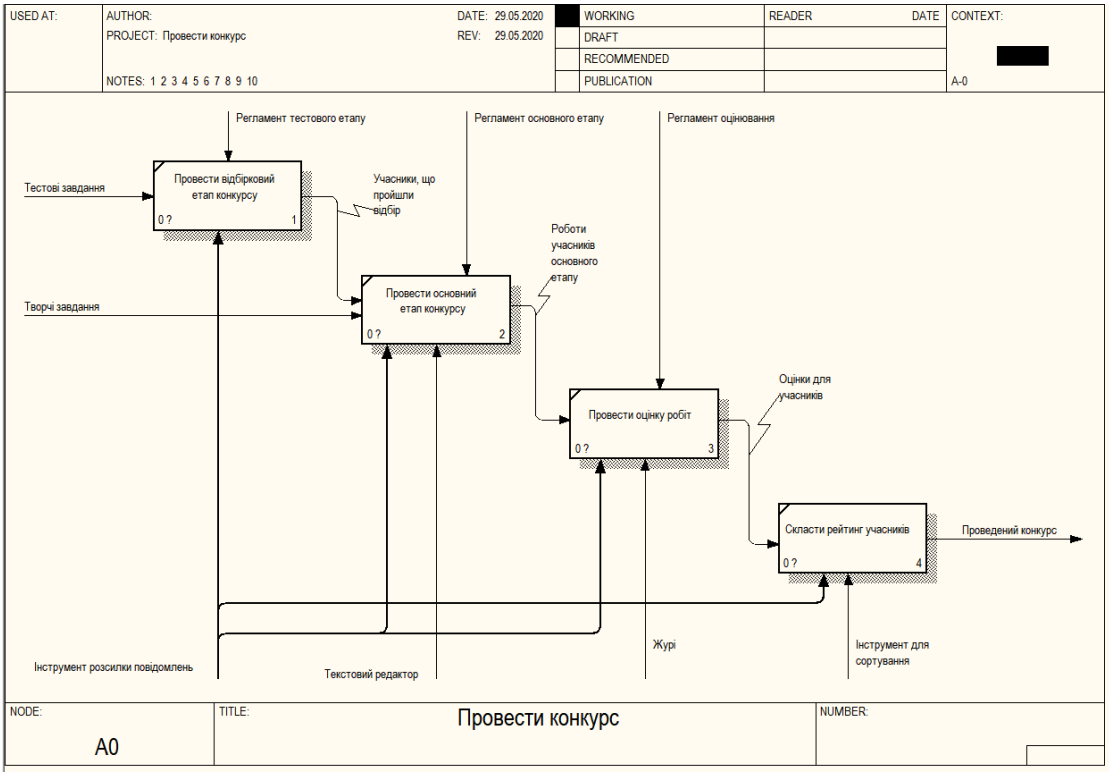


Рисунок 1.2 – Схема структурно-функціональна декомпозиції процесу проведення конкурсу в нотаціях IDEF0

Як показано на рисунку 1.2, проведення конкурсів можна декомпозувати на 4 основних функції, а саме: «провести відбірковий етап конкурсу», «провести основний етап конкурсу», «провести оцінку робіт» та «скласти рейтинг учасників». Для проведення відбіркового етапу необхідні вхідні дані у вигляді тестових завдань. Відбірковий етап регулюється регламентом тестового етапу. Необхідний механізм проведення відбіркового етапу – інструмент розсилки повідомлень.

Для виконання функції проведення основного етапу необхідні вхідні дані у вигляді творчих завдань і учасників, що пройшли відбір. Основний етап регулюється регламентом проведення основного етапу, а механізмами виступають інструмент розсилки повідомлень і текстовий редактор.

Для виконання функції проведення оцінки робіт необхідні вхідні дані у вигляді робіт учасників. Регулюється функція регламентом проведення оцінювання. Механізмами слугують журі і інструмент розсилки повідомлень.

Для виконання функції складання рейтингу учасників необхідно мати вхідні дані у вигляді оцінок робіт учасників. Необхідні механізми для складання рейтингу – інструмент розсилкі повідомлень і інструмент сортування.

1.1.1 Опис процесу діяльності

В процесі організації і проведенні творчих письмових конкурсів виявлені такі проблеми:

Оскільки конкурси не проводяться по якомусь одному заданому шаблону, програмне забезпечення для проведення таких конкурсів мало б надавати усім групам акторів, що з ним взаємодіють, гнучкі інструменти для налаштування регламенту проведення етапів конкурсів, перегляду стану проведення конкурсу на даний конкретний момент та оцінювання конкурсних робіт, щоб задовільнити як можна ширший діапазон потреб користувачів та водночас створити централізовану систему, яка покрила б усі потреби кінцевих користувачів.

На вибір саме такої теми дипломного проекту також вплинула наявність даних про проведення різних конкурсів, надана організаторами цих конкурсів. Конкурси проводились децентралізовано, тобто, за різні етапи проведення конкурсів відповідали різні програмні забезпечення в зв'язку з чим в процесі проведення виникали складності з обміном роботами між учасниками та суддями, виставленням оцінок (яке ніяк не було автоматизовано і рейтинг доводилось складати вручну).

На рисунку 1.3 показано дерево проблем.

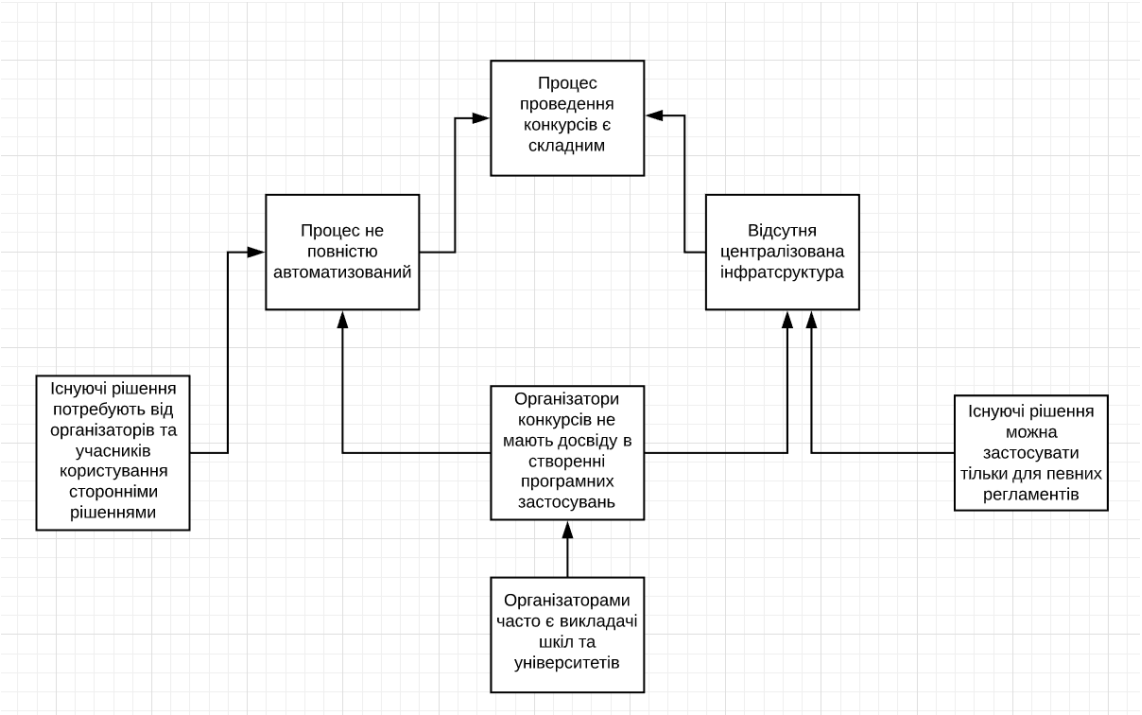


Рисунок 1.3 – Дерево проблем

На рисунку 1.4 надана схема структурна діяльності на поточний момент.

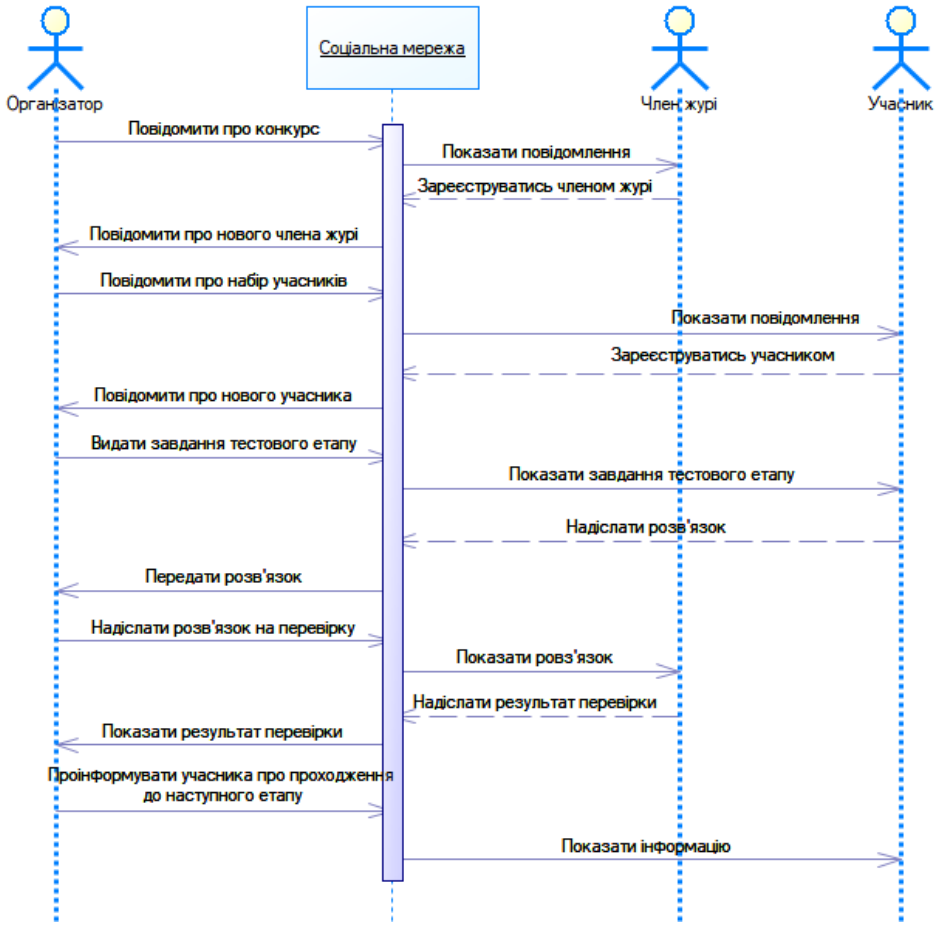


Рисунок 1.4 – Схема структурна діяльності на поточний момент

1.1.2 Опис функціональної моделі

Актори – «організатор», «журі», «учасник».

Актор «організатор» може надавати завдання у вигляді текстів тестових завдань та варіантів відповідей, а також текстів творчих завдань, та визначати регламент проведення конкурсу, а саме систему оцінювання тестів, систему проведення основного етапу та порядок оцінювання робіт учасників членами журі. Актор «журі» може оцінити роз'язок завдання за умови, що актор «учасник» надішле розв'язок і актор «організатор» визначить систему балів за якою необхідно оцінити завдання. Актор «учасник» може отримати завдання, переглянути свою оцінку та надіслати розв'язок. Окрім цього усі актори можуть слідкувати за поточним станом конкурсу.

Для того, щоб кожен актор міг виконувати тільки функціонал, що передбачений саме для нього, необхідно на початку роботи програмного забезпечення проводити авторизацію, і тому усім акторам має бути надана можливість увійти в систему.

На рисунку 1.5 наведена діаграма варіантів використання.

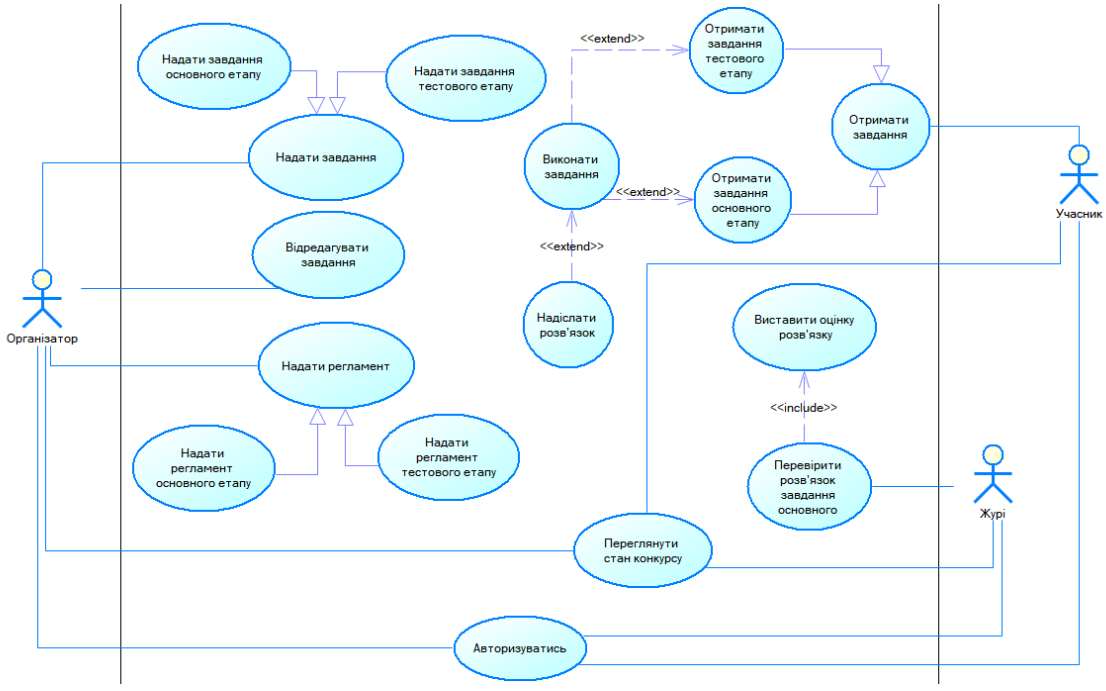


Рисунок 1.5 – Діаграма варіантів використання

1.2 Огляд наявних аналогів

На момент написання звіту відомих автоматизованих систем проведення творчих (чи інших) видів конкурсів у відкритому доступі чи на платній основі знайти не було можливо. Творчі конкурси прийнято проводити або в форматі «на місці», тобто учасники мають з’явитися на місце проведення конкурсу та очно виконувати свою роботу, або ж з використанням веб-сайту для проведення конкретного конкурсу, котрий надає інтерфейс лише для завантаження учасниками їх робіт і не може бути використаний для проведення інших конкурсів, регламент яких має незначні відмінності[3].

Значний недолік таких систем проведення конкурсів полягає у тому, що вони не централізують усі етапи проведення конкурсу і не надають усім користувачам необхідні інтерфейси для участі і проведення конкурсів, а лише приховують спілкування між учасниками та організаторами, і вже на плечі останніх кладеться задача взаємодії з журі і підбиття підсумків проведення конкурсу.

Також популярною є система проведення конкурсу через соціальні мережі, коли за допомогою можливостей особистих повідомлень та, в окремих випадках, месенджер-ботів відбувається спілкування між сторонами системи[2].

Недолік такої системи полягає в повній відкритості соціальної мережі, тобто, за бажанням кожен учасник може визначити, хто був у суддівському журі та хто був організатором. Така система не є раціональною з точки зору безпеки користувачів.

В таблиці 1.1 зведена інформація про наявність аналогів.

Таблиця 1.1 – Аналоги систем проведення конкурсу

Назва	**Централізо-ваність**	**Можливість автоматично видавати завдання**	**Можливість автоматично формувати рейтинг**	**Можливість обирати різні сценарії проведення**
Соціальні мережі	-	+	-	+
Веб-сайт конкурсу	+	+	+	-
Боти	-	+	+	-

1.3 Постановка задачі

1.3.1 Призначення розробки

Система призначена для автоматизації процесу організації конкурсів, що передбачають взаємодію учасників та журі.

1.3.2 Цілі та задачі розробки

Ціль: покращити організацію, процеси проведення і оцінювання конкурсних творчих робіт за рахунок автоматизації видачі завдань та їх оцінювання.

Для досягнення цілі необхідно розв'язати наступні задачі:

- визначити функції системи;
- створити інформаційне забезпечення у вигляді масивів даних;
- розробити архітектуру системи;
- спроектувати інтерфейс;

– мінімізувати кількість можливих помилок шляхом тестування програмного продукту.

Висновок до розділу

У даному розділі було описано предметне середовище, у якому функціонує система, та досліджено проблеми, що постають перед розробкою. Для вирішення проблем було поставлено задачу до розробки, визначено цілі розробки, побудовано діаграму варіантів використання, в якій розглянуто взаємодію акторів з системою, розглянуто існуючі аналоги серед представлених в мережі Інтернет і проведено порівняльний аналіз цих аналогів. Окрім того, було виявлено призначення розробки.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

На вхід системі подаються:

- дані авторизації користувачів;
- відповіді на тести;
- відповіді користувачів на конкурсні завдання;
- оцінка журі.

Реквізити наведені в таблиці 1.1:

Таблиця 2.1 – Реквізити для вхідних даних

№	Найменування	Реквізити
1	Дані авторизації користувачів	Логін, пароль, група користувача
2	Відповіді на тести	Вибрані варіанти відповідей
3	Відповіді на конкурсні завдання	Тексти відповідей на конкурсні завдання
4	Оцінка журі	Бали за виконане завдання, оцінка складності тестового завдання

2.2 Вихідні дані

На виході система надає:

- тестове завдання;
- завдання основного етапу;
- рейтингове місце учасника.

Реквізити наведені в таблиці 1.2.

Таблиця 2.2 – Реквізити вихідних даних

№	Найменування	Реквізити
1	Тестове завдання	Текст завдання, текст варіантів відповідей
2	Завдання основного етапу	Текст завдання, поле вводу розв'язку
3	Рейтингове місце учасника	Рейтинговий список усіх учасників, місце конкретного учасника

2.3 Структура масивів інформації

Інформація про користувачів та етапи зберігається в окремих mdf-файлах, оскільки використання баз даних сповільнило б роботу програмного забезпечення, а виграшу під час розробки від баз даних немає.

Структура файлів приведена в таблиці 2.3.

Таблиця 2.3 – Структура масивів інформації

№	Файл	Склад
1	users.mdf	Перше поле – кількість користувачів n. Далі інформація про n користувачів з нового рядка: - логін - пароль (SHA-256 хеш) - група

Продовження таблиці 2.3

2	contests.mdf	Перше поле – кількість конкурсів n. Далі інформація про n конкурсів з нового рядка: - учасники; - кількість етапів; - опис етапів; - регламент оцінок.
3	works.mdf	Перше поле – кількість робіт n. Далі інформація про n робіт з нового рядка: - логін учасника, що надіслав роботу; - ідентифікатор конкурсу, до якого надіслана робота; - оцінки від членів журі.

Висновок до розділу

Описано інформаційне забезпечення системи. Було розроблено структуру масивів інформації, наведено реквізити вхідних та вихідних дані інформаційної системи. Розроблено бекенд частину системи.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задачі

Найсуттєвіша математична задача, що ставиться в розробці проєкту, – це побудова рейтингу учасників конкурсу за результатами проведення тестового завдання, якщо таке передбачено регламентом.

3.2 Математична постановка задачі

Нехай є N учасників та M тестових запитань. Про кожне запитання j, $j \in 1 \dots M$, відомо, чи відповів на нього учасник i, $i \in 1 \dots N$. Задача полягає у тому, щоб по відповідям учасників побудувати рейтинг учасників так, щоб забезпечити найменшу кількість колізій, тобто, мінімізувати кількість учасників, що мають однаковий конкурсний бал.

3.3 Обґрунтування методу розв’язання

Для заданої постановки задачі існує ряд методів розв'язання [4]. Перший метод – створити більше ніж n питань. За кожну правильно відповідь на тестове запитання нараховувати учаснику 1 бал, і, оскільки кількість тестів більша за кількість учасників, кожен учасник зможе набрати унікальну рейтингову оцінку. Але можливо, що два учасники наберуть однакову суму балів, при цьому відповівши на різні запитання. У такому випадку їх конкурсні бали будуть ідентичними. Окрім того, процес написання гарних запитань є тривалим, і створити велику кількість гарних запитань не є можливим[5,6].

Тривіальне рішення проблеми полягає у тому, щоб за різні тестові запитання нараховувати різну кількість тестових балів. Тоді, якщо 2 учасники відповіли на різні питання, вони отримають різні бали. Але все ще можна виділити такі два набори відповідей учасників, що сума балів за відповіді буде рівна.

Тому пропонується наступний принцип формування тестового рейтингу.

– Перед початком тестування, організатор призначає за кожне запитання суму балів від 1 до l.
– Для кожного питання обраховується кількість учасників, що відповіли на нього.
– Значення, отримані з попередніх двох пунктів, приводяться до однієї шкали і множаться, щоб визначити результуючий бал учасника за завдання.

При такому підході кількість колізій буде зведено до мінімуму.

3.4 Опис методів розв'язання

Нехай в системі є 4 учасники та 3 тестових питання. Кожен з учасників відповів на 2 запитання. Якщо оцінювати кожне запитання в 1 бал, то всі четверо учасників будуть мати однаковий рейтинговий бал. Але кожний учасник міг відповісти на різні пари запитань, оскільки варіантів вибору двох відповідей з трьох пронумерованих запитань

$$A_3^2 = \frac{3!}{2!} = 3. \quad (3.1)$$

Для того, щоб учасники могли отримати різні рейтингові бали, вводяться різні суми балів за різні запитання, наприклад, за перше питання учасники отримають 8 балів, за друге – 7 і за третє – 2. Розглянемо приклад, коли учасники відповіли на запитання наступним чином:

Перший учасник: перше та друге питання.

Другий учасник: перше та друге питання.

Третій учасник: перше та третє питання.

Четвертий учасник: друге та третє питання.

У відповідності до запропонованих балів за запитання, буде складено такий рейтинг учасників:

Перше і друге місця: учасники з номерами 1 і 2, по 15 балів.

Третє місце: учасник з номером 3, 10 балів.

Четверте місце: учасник 4, 9 балів.

Тепер за відповіді на різни запитання учасники отримують різну кількість балів. Але при цьому учасник 4, що відповів на запитання 2 і 3, на які відповіла менша кількість учасників, ніж на питання 1, посів останнє місце, оскільки питання були оцінені меншою кількістю балів. Далі приведений розподіл відповідей на запитання:

1 запитання – 3 учасники

2 запитання – 3 учасники

3 запитання – 2 учасники

З цієї інформації можна зробити висновок, що питання 3 виявилося складнішим за питання 1 і 2, і тому варто за нього нараховувати пропорційно більшу кількість балів.

Якщо за кожне питання нараховуватиметься величина балів, що становить:

$$f_{ij} = \frac{1}{m_j}, \tag{3.2}$$

де m_j – кількість відповідей на запитання з номером i,

тоді запитання будуть оцінені наступними балами:

$f_1 = 0{,}33,$

$f_2 = 0{,}33,$

$f_3 = 0{,}5.$

Сума балів учасника за такою системою вираховується за формулою

$$S_i^1 = \sum_{j=1}^{M} \frac{r_{ij}}{m_j} = \sum_{j=1}^{M} r_{ij} f_{ij}, \tag{3.3}$$

де $r_{ij} = \begin{cases} 1, \text{ якщо учасник } i \text{ відповів на питання } j \\ 0, \text{якщо учасник } i \text{ невідповів на питання } j \end{cases}$ (3.4)

Тоді буде отримано наступний рейтинг учасників:

Перше і друге місце – учасники з номерами 3 і 4, $S_3^1 = S_4^1 = 0{,}83$ балу

Третє і четверте місце: учасники 1 і 2, $S_1^1 = S_2^1 = 0{,}66$ балу

Цей варіант системи винагороджує учасників, що відповіли на статистично найскладніші питання серед представлених у тесті.

На рисунку 3.1 показано ранжування, побудоване двома описаними вище методами на основі даних про проходження 20 тестів учасниками реального конкурсу.

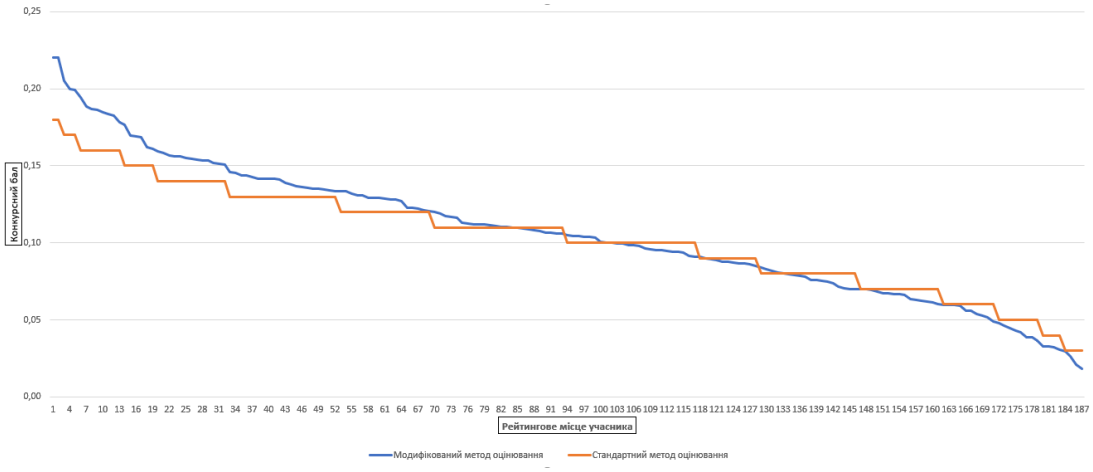


Рисунок 3.1 – Порівняння результатів розрахунків двох методів

Помаранчева лінія – метод, коли за кожну відповідь нараховується 0,01 бал. Синя – коли сума балів визначається за формулою (3.3). Учасників 187, максимальна кількість вірних відповідей – 18.

Прямі відрізки на графіку свідчать, що учасники з цього відрізка отримали рівну кількість балів. Таких відрізків, при використанні першого методу, дуже багато, через що неможливо точно визначити, яке місце посів який учасник. Синій графік майже не має прямих відрізків, а отже метод добре нівелює колізії.

Можливо також формувати рейтинг одночасно на основі кількості учасників, що відповіли на запитання, та суми балів, яку організатори визначили за запитання. Необхідно виконати нормування оцінок організаторів до шкалі від 0 до 1, оскільки в шкалі від 0 до 1 знаходяться оцінки, що отримані за формулою (3.3). Вводиться наступна формула розрахунку коефіцієнтів поправки:

$$q_j = \frac{z_j}{l}, \tag{3.5}$$

де q – шуканий коефіцієнт,

z_j – оцінка за запитання j за шкалою від 1 до l:

$$q_1 = 0{,}8$$

$$q_2 = 0{,}7$$

$$q_3 = 0{,}3.$$

Наступний крок - виконання операції множення оцінки за запитання на отримані коефіцієнти, результатом якого стануть оцінки з призначеними вагами складності запитань, визначені за формулою

$$w_j = \frac{z_j}{lm_j}. \tag{3.6}$$

$$w_1 = 0{,}264,$$

$$w_2 = 0{,}231,$$

$$w_3 = 0{,}15.$$

Рейтинг буде сформовано за параметром, визначеним наступною формулою:

$$S_i^2 = \sum_{j=1}^{M} r_{ij} w_j. \tag{3.7}$$

Рейтинг за параметром (3.7) має наступний вигляд:

- перше і друге місце – учасники з номерами 1 і 2, $S_1^2 = S_2^2 = 0{,}495$;
- третє місце – учасник з номером 3, $S_3^2 = 0{,}414$ балу;
- четверте місце – учасник з номером 4, $S_4^2 = 0{,}381$ балу.

Така система дозволяє врахувати складність запитань одразу за двома критеріями: оцінкою від організатора та оцінкою, що отримана на основі розподілу відповідей на запитання. Шкала оцінок є достатньо широкою (в розглянутому вище варіанті оцінки мають 3 знаки після коми, тобто є достатніми для ранжування 10000 учасників за умови достатньої кількості тестових запитань).

Висновок до розділу

Було побудовано математичну модель проведення тестового етапу конкурсу. Модель порівняно з тривіальними і визначено її переваги. Побудовано систему ранжування за якої в рейтинговій таблиці буде якнайменше колізій.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Система розроблялась за допомогою інструментів, що представлені мовою програмування Java 11, оскільки вони мають необхідний і достатній функціонал для виконання усіх задач і досягнення усіх цілей, що поставлені перед розробкою [7].

Для розробки графічного інтерфейсу користувача було використано інструмент JFrame з бібліотеки java.swing, оскільки в ньому містяться зручні інструменти для налаштування інтерфейсу[8], розміщення його елементів та розробки його функцій, що дозволяє пришвидшити процес розробки застосунку при цьому підвищуючи якість вихідного продукту шляхом уникнення помилок, які могли б виникнути, якщо інструмент використаним би не був [9].

Для побудови системи на основі клієнт-серверної архітектури застосунку було використано Networking-можливості, що також представлені в Java 11, зокрема Socket і ServerSocket, які можуть бути використані для налаштування з'єднання між клієнтом та сервером, і вхідні та вихідні потоки, реалізовані за допомогою Thread і Runnable об'єктів, які створюються Socket і ServerSocket для моделювання спілкування між клієнтом та сервером, тобто обміну повідомлень.

4.2 Вимоги до технічного забезпечення

Мінімальні вимоги для технічного забезпечення клієнту:

- оперативна пам'ять – 512 Мб;
- місткість жорсткого диску – 1 Гб;
- процесор – одноядерний, 1 ГГц;
- монітор, HD-графічна карта.

Рекомендовані вимоги для технічного забезпечення клієнту:

- оперативна пам'ять – 2 Гб;
- місткість жорсткого диску – 1 Гб;
- процесор – двоядерний, 1 ГГц;

Монітор, HD-графічна карта.

Мінімальні вимоги для технічного забезпечення серверу:

- оперативна пам'ять – 1 Гб;
- місткість жорсткого диску – 5 Гб;
- процесор – двоядерний, 1 ГГц.

Рекомендовані вимоги для технічного забезпечення серверу:

- оперативна пам'ять – 4 Гб;
- місткість жорсткого диску – 20 Гб;
- процесор – восьмиядерний, 3 ГГц.

4.2.1 Загальні вимоги

4.2.1.1 Вимоги до функціональних характеристик

Актор «організатор» може надавати завдання та визначати регламент проведення конкурсу. Актор «журі» може оцінити роз’язок завдання за умови, що актор «учасник» надішле розв’язок і актор «організатор» визначить систему балів за якою необхідно оцінити завдання. Актор «учасник» може отримати завдання, переглянути свою оцінку та надіслати розв’язок.

Для того, щоб кожен актор міг виконувати тільки функціонал, що передбачений саме для нього, необхідно на початку роботи програмного забезпечення проводити авторизацію, і тому усім акторам має бути надана можливість увійти в систему.

4.2.1.2 Вимоги до надійності

Система повинна функціонувати безвідмовно незважаючи на наявність ймовірних дефектів, які можуть проявлятися під час експлуатації.

Виправлення таких дефектів повинно виконуватися на етапах розробки, бета-тестування та в процесі введення в дію і в межах промислової експлуатації.

Відмова програмного забезпечення сервісу не повинна призводити до руйнувань даних в інформаційних сховищах.

Будь-які аварійні ситуації повинні бути негайно задокументовані і надіслані розробникам задля усунення причин збою в роботі системи.

4.3 Архітектура програмного забезпечення

4.3.1 Діаграма класів

На діаграмі класів серверної частини застосунку, що приведена на рисунку 4.1, показано взаємодію класів програмного продукту, що виконують взаємодію на рівні серверу

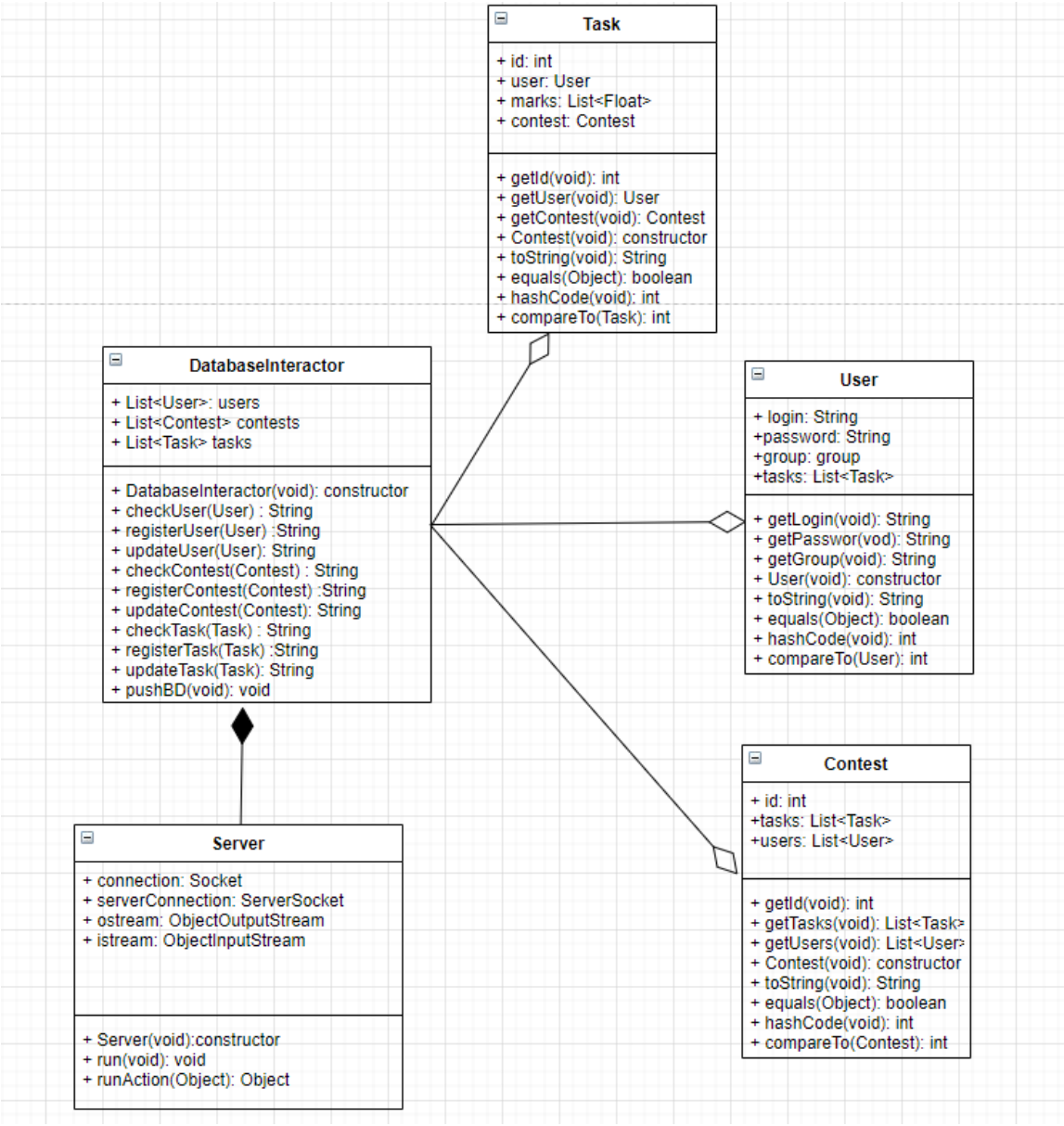


Рисунок 4.1 – Діаграма класів серверної частини

На діаграмі класів серверної частини застосунку показано взаємодію класів програмного продукту, що виконують взаємодію на рівні серверу та забезпечують роботу за шаблоном Model-View-Controller для представлення та збереження інформації і взаємодії між представленням і інформаційними моделями. Моделями є дані користувачів, конкурсів і завдань, сервер представляє інформацію, а контролером є клас DatabaseInterractor, що дозволяє взаємодіяти моделі і представленню.

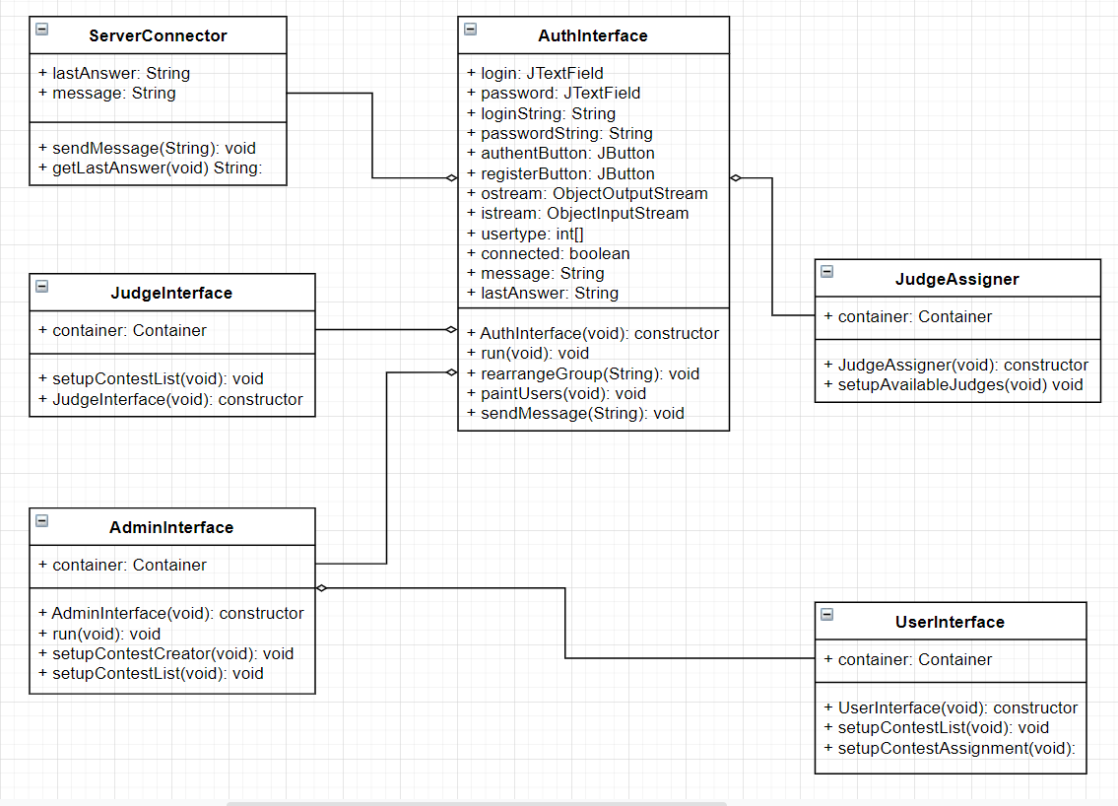


Рисунок 4.2 – Діаграма класів клієнтської частини

На діаграмі класів клієнтської частини показано, як взаємопов'язані між собою класи, що відповідають за роботу клієнта (рисунок 4.2). Основна частина логіки описана класами AuthInterface та ServerConnector, а інші класи виконують функції моделей для графічного інтерфейсу.

4.3.2 Діаграма послідовності

На діаграмі послідовності, що приведена на рисунку 4.3 продемонстрований процес взаємодії акторів з об'єктами системи і показаний процеж життєдіяльності об'єктів.

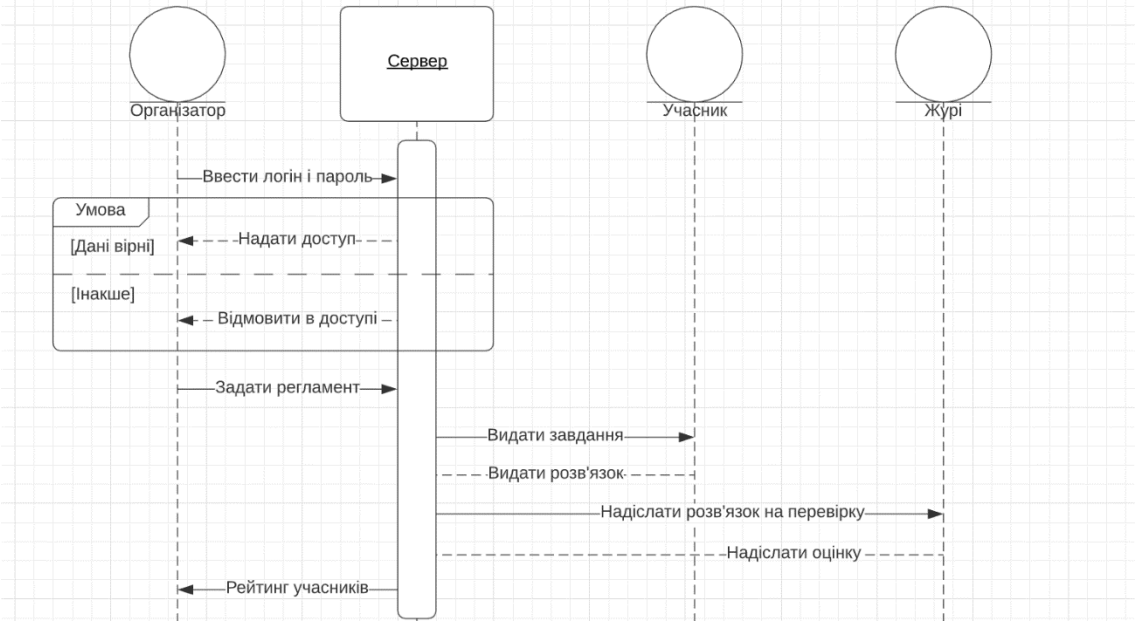


Рисунок 4.3 – Діаграма послідовності

Як показано на діаграмі, сервер існує весь час роботи програми і в першу чергу слугує для надання доступу користувачам до системи та обміну повідомленнями, а саме завданнями та розв'язками, що показано під частиною з авторизацією і заданням регламенту.

Показано, що актори взаємодіють виключно через сервер і не ведуть прямого контакту, тобто ціль автоматизації видачі завдань та їх перевірки досягається успішно.

Завдяки перевірці даних користувачі не можуть отримувати доступ до даних, що їм не призначені, тобто не можуть побачити готові відповіді або роботи інших учасників.

4.3.3 Діаграма компонентів

Діаграма компонентів системи приведена на рисунку 4.4.

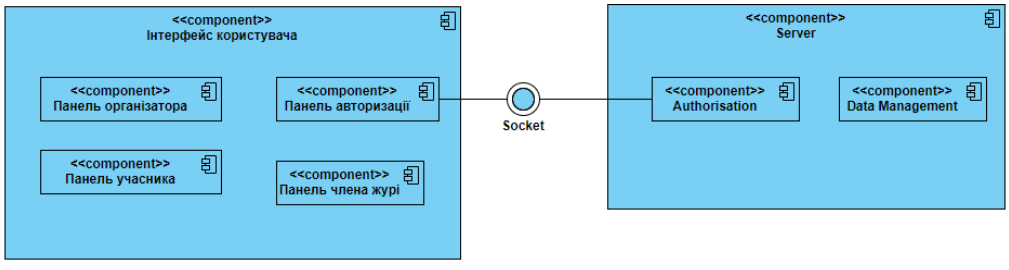


Рисунок 4.4 – Діаграма компонентів

Основні компоненти даної інформаційної системи – це сервер і користувацький інтерфейс, які в свою чергу поділені на відповідні підкомпоненти, а саме для інтерфейсу користувача це панель організатора, панель авторизації, панель учасника та панель члена журі, які проедставляють функціонал яким користуються відповідні актори системи, а серверний компонент, через який інтерфейсний компонент пов'язаний інтерфейсом Socket, відповідає за автентифікацію та управління даними системи, такими як дані користувача.

4.3.4 Схема алгоритму

На рисунку 4.5 приведено схему роботи алгоритму.

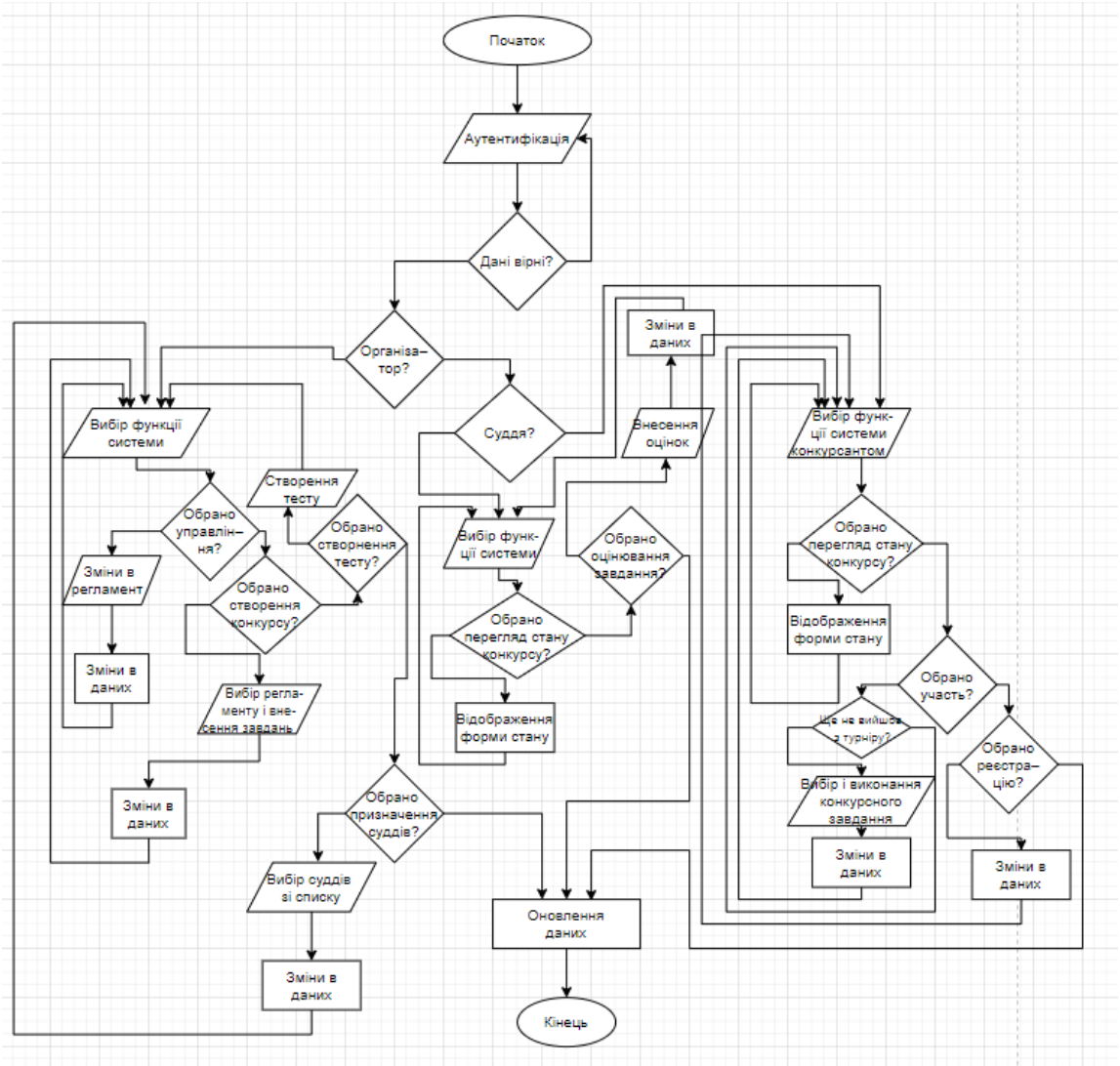


Рисунок 4.5 – Схема алгоритму

4.3.5 Специфікація функцій

В таблиці 4.1 приведена специфікація функцій програмного продукту.

Таблиця 4.1 – Специфікація функцій

Назва функції	**Клас**	**Опис дії**
public void sendMessage (String message)	ServerConnector	Надсилає серверу сформоване клієнтом повідомлення
public String getLastAnswer()	ServerConnector	Надає останню відповідь від сервера
public void run()	AuthInterface	Відповідає за управління потоком класу та загальні виклики методів взаємодії з сервером

Продовження таблиці 4.1

Назва функції	**Клас**	**Опис дії**
private void rearrangeGroup(String group)	AuthInterface	У відповідності з отриманою від сервера групою користувача готує виклик форми, що відповідає отриманій групі
private void sendMessage(String message)	AuthInterface	Формує повідомлення, що потім буде відправлено класом ServerConnector
public String checkUser(User user)	DatabaseInterractor	Надає запит до масиву даних на перевірку наявності у ньому користувача з отриманими авторизаційними даними та відповідає групою користувача чи помилкою

Продовження таблиці 4.1

Назва функції	**Клас**	**Опис дії**
public void registerUser(User user)	DatabaseInterractor	Перевіряє на наявність у масивах даних користувача з таким іменем та у разі відсутності додає до масиву відповідну інформацію та повертає групу користувача або помилку
private void pushBD()	DatabaseInterractor	Виконує запит на додавання даних до масиву даних
public void run()	Server	Відповідає за організацію роботи потоку сервера та виконує загальні виклики серверних функцій
public Object runAction(Object action)	Server	Обробляє заголовки клієнтських запитів та викликає функції взаємодії з масивами даних
public String getGroup()	User	Геттер для групи користувача

Продовження таблиці 4.1

Назва функції	**Клас**	**Опис дії**
public String getLogin()	User	Геттер для логіну користувача
public String getPassword()	User	Геттер для зашифрованого паролю користувача
public String toString()	User	Метод переведення інформації про користувача в рядковий тип даних
public boolean equals(Object o)	User	Метод перевірки двох користувачів на еквівалентність
public int hashCode()	User	Метод формування хешу
public int compareTo(User user)	User	Метод порівняння двох користувачів для сортування
public void setupContestCreator()	AdminInterface	Слугує для розташування та опису логіки елементів, що пов'язані зі створенням конкурсів

Продовження таблиці 4.1

Назва функції	**Клас**	**Опис дії**
public void setupContestList()	AdminInterface	Слугує для розташування та опису логіки елементів, що пов'язані з відображенням активних конкурсів
private void setupAvailableJudges()	JudgeAssigner	Слугує для розташування та опису логіки елементів, що пов'язані з відображенням суддів, яких можна призначити на конкурс
public void setupContestList()	JudgeInterface	Слугує для розташування та опису логіки елементів, що пов'язані з відображенням активних конкурсів
protected void setupContestList()	UserInterface	Слугує для розташування та опису логіки елементів, що пов'язані з відображенням активних конкурсів

Продовження таблиці 4.1

Назва функції	**Клас**	**Опис дії**
protected void setupContestAssignment()	UserInterface	Слугує для розташування та опису логіки елементів, що пов'язані з відображенням конкурсів, у яких користувач приймає участь

Висновок до розділу

В розділі було побудовано діаграми класів, процесів та компонентів, що дозволяють розглянути інформаційну систему з точки зору проектування та запобігти написанню неструктурованого та неоптимального коду під час реалізації за рахунок використання паттернів, які вважаються прикладами написання гарного коду. Було описано специфікацію функцій класів системи, за допомогою якої простіше зрозуміти функціональну структуру класів.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

На початку роботи програми користувачу виводиться форма авторизації у якій він може або спробувати авторизуватись, використавши при цьому раніше зареєстровані в системі дані (факт реєстрації буде перевірено сервером) або зареєструватись як новий користувач, при цьому логін і пароль, що користувач ввів в форму авторизації будуть використані системою у якості його даних авторизації.

Форма авторизації приведена на рисунку 5.1:

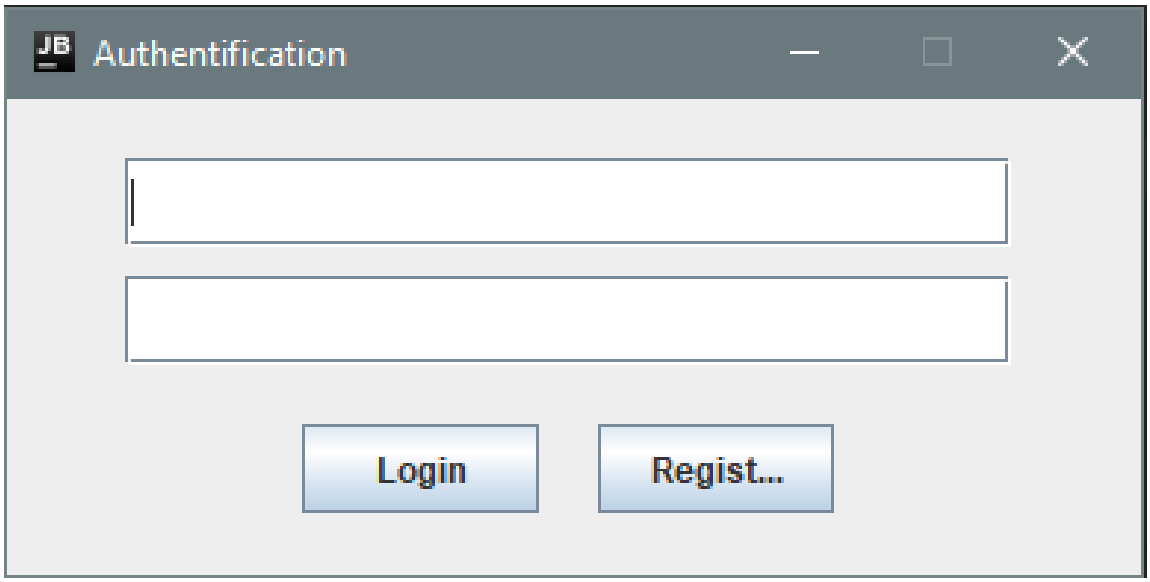


Рисунок 5.1 – Форма авторизації

Після проходження процесу авторизації або реєстрації користувачу виводиться форма, що відповідає його групі користувача. Якщо користувач зареєструвався щойно, то йому буде надана група «учасники».

Форма для організаторів конкурсів приведена на рисунку 5.2:

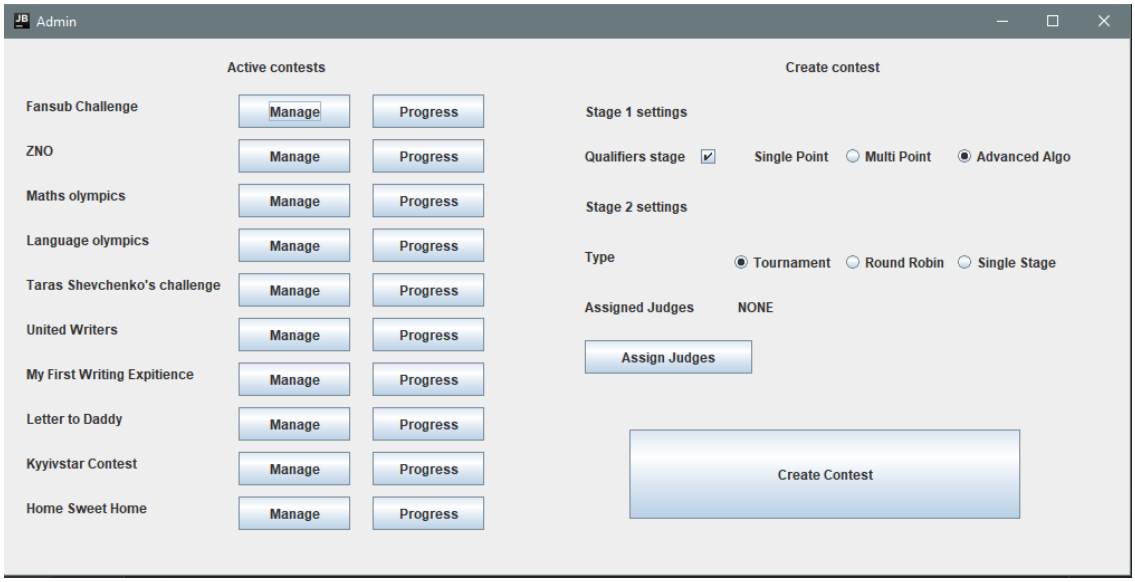


Рисунок 5.2 – Інтерфейс організаторів

Після натискання на пункт меню «Assign Judges» користувачу надається можливість призначити суддів за допомогою відповідної форми. За замовчуванням кожен суддя відмічений як «Not Assigned», тобто не призначений. Відмітивши чекбокс «Assigned» (призначений) напроти відповідного судді, користувач цією дією призначає суддю до створюваного конкурсу. Після закриття цієї форми відмічені чекбокси переносяться до попередньої з відповідними змінами.

Форма призначення суддів приведена на рисунку5.3:

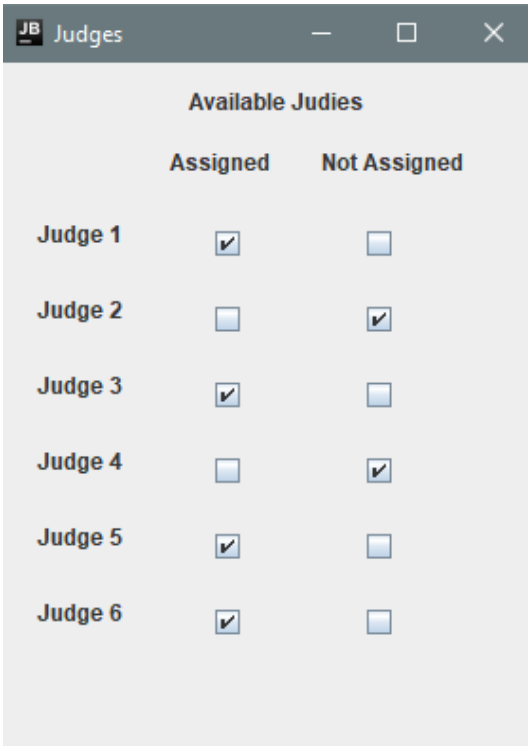


Рисунок 5.3 – Інтерфейс призначення суддів

Інтерфейс для учасників схожий на інтерфейс адміністратора, але замість кнопки для управління є кнопка для реєстрації на конкурсі. Також справа виводиться інформація про всі конкурси, на які користувач на даний момент зареєстрований.

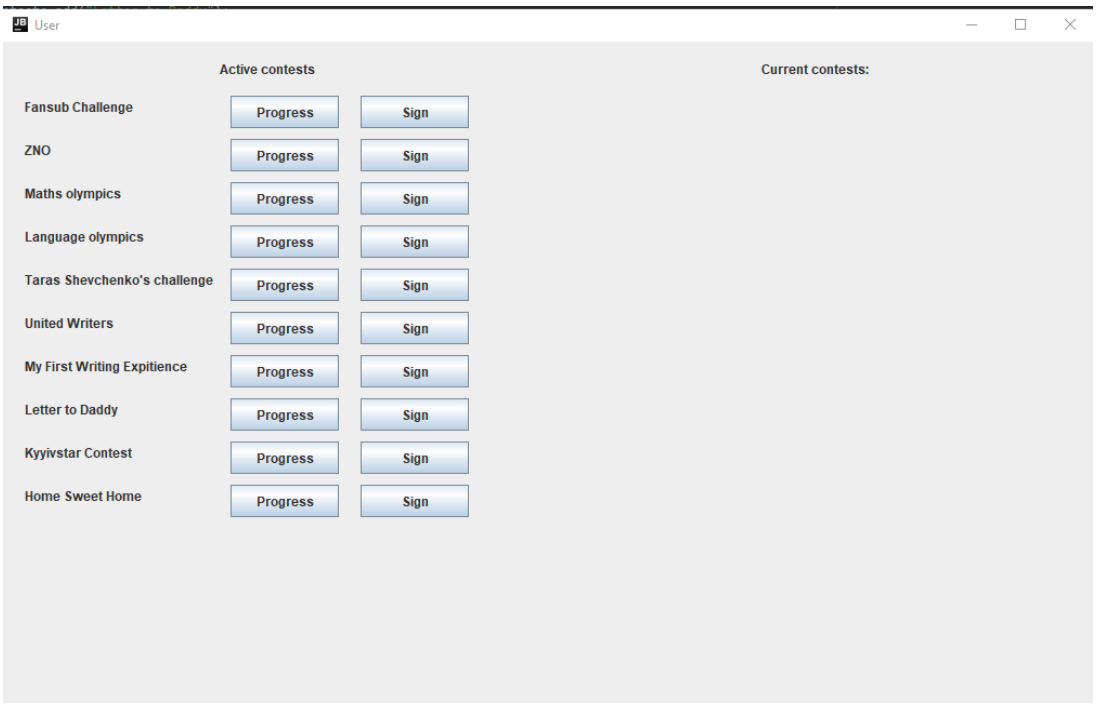


Рисунок 5.4 – Інтерфейс учасників

Інтерфейс для журі передбачає відображення усіх активних конкурсів, до яких член журі був призначений організаторами. В нього є можливість подивитись хід конкурсу а також відкрити інтерфейс для оцінювання даного конкурсу.

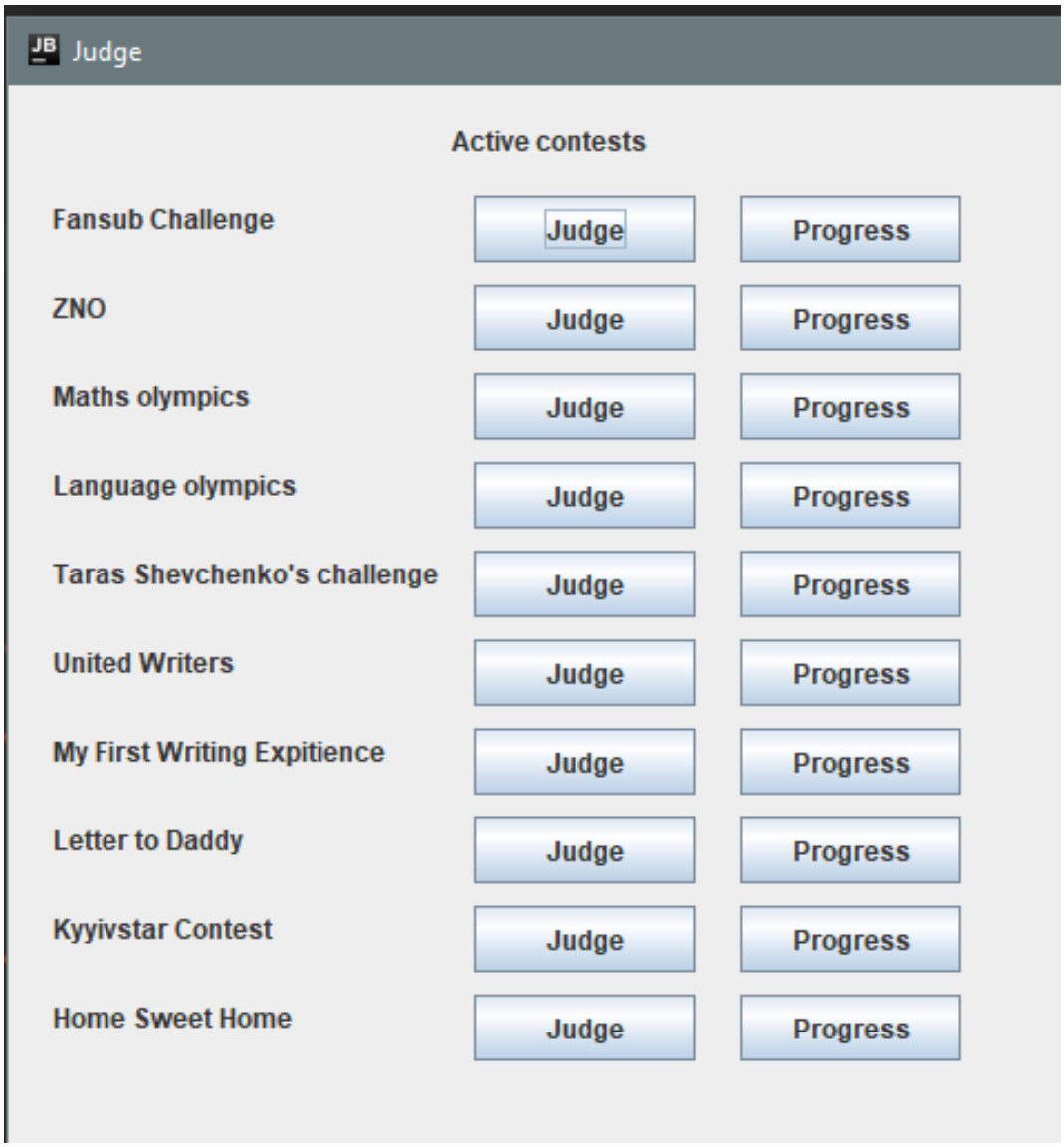


Рисунок 5.5 – Інтерфейс суддів

5.2 Випробування програмного продукту

5.2.1 Мета випробувань

Метою випробувань є перевірка відповідності функцій комплексу задач підтримки проведення творчих письмових конкурсів вимогам технічного

завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;
– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

В процесі тестування була перевірена уся функціональність комплексу задач (КЗ). У наступних таблицях наведений перелік випробувань основних функціональних можливостей та результати, що свідчать про відповідність чи невідповідність функцій вимогам.

Таблиця 5.1 – Результати перевірки функції авторизації

Мета тесту	Перевірка функції авторизації
Початковий стан КЗ	Відкрите вікно авторизації
Вхідні данні	Логін та пароль користувача
Схема проведення тесту	Ввести логін і пароль у відповідні поля
Очікуваний результат	Відкрита форма групи користувача
Стан КЗ після проведення випробувань	Відкрита форма групи користувача

Таблиця 5.2 – Результати перевірки функції створення нового конкурсу

Мета тесту	Перевірка функції створення нового конкурсу
Початковий стан КЗ	Відкрита панель адміністратора
Вхідні данні	Регламент конкурсу

Мета тесту	Перевірка функції створення нового конкурсу
Схема проведення тесту	За допомогою графічного інтерфейсу визначити систему проведення конкурсу
Очікуваний результат	Створено конкурс з заданим регламентом
Стан КЗ після проведення випробувань	Створено конкурс з заданим регламентом

Таблиця 5.3 – Результати перевірки функції призначення журі

Мета тесту	Перевірка функції призначення журі
Початковий стан КЗ	Відкрита панель призначення журі
Вхідні данні	Члени журі
Схема проведення тесту	За допомогою графічного інтерфейсу обрати членів журі, яких буде призначено до конкурсу
Очікуваний результат	В меню створення та управління конкурсом вказано призначених до нього членів журі
Стан КЗ після проведення випробувань	В меню створення та управління конкурсом вказано призначених до нього членів журі

Таблиця 5.4 – Результати перевірки функції перегляду стану конкурсу

Мета тесту	Перевірка функції перегляду стану конкурсу
Початковий стан КЗ	Відкрита панель стану конкурсу
Вхідні данні	—

Продовження таблиці 5.4

Схема проведення тесту	Перевірити виведені дані на відповідність актуальній інформації
Очікуваний результат	На панелі відображено актуальний стан конкурсу
Стан КЗ після проведення випробувань	На панелі відображено актуальний стан конкурсу

Таблиця 5.5 – Результати перевірки функції реєстрації

Мета тесту	Перевірка функції реєстрації
Початковий стан КЗ	Відкрите вікно авторизації
Вхідні данні	Логін та пароль користувача
Схема проведення тесту	Ввести логін і пароль у відповідні поля та натиснути кнопку «Register»
Очікуваний результат	Відкрита форма для учасника конкурсів
Стан КЗ після проведення випробувань	Відкрита форма для учасника конкурсів

Таблиця 5.6 – Результати перевірки функції реєстрації учасника конкурсу

Мета тесту	Перевірка функції реєстрації учасника конкурсу
Початковий стан КЗ	Відкрита панель користувача
Вхідні данні	Обраний конкурс

Продовження таблиці 5.6

Схема проведення тесту	Натиснути на кнопку «Sign»
Очікуваний результат	Повідомлення про успішну реєстрацію
Стан КЗ після проведення випробувань	Повідомлення про успішну реєстрацію

Висновок до розділу

В розділі було надано розроблене керівництво користувача, в якому представлені функції, що забезпечені системою у відповідності до постановки задачі, наведеної в першому розділі, та приведено рисунки відповідних графічних інтерфейсів та форм.

Було проведено процес тестування програмного забезпечення і встановлено, що програмне забезпечення виконує усі заявлені функції у відповідності до передбачених результатів, що є бажаними з точки зору роботоздатності.

ЗАГАЛЬНІ ВИСНОВКИ

В дипломному проекті було описано та розроблено інформаційну систему, що забезпечує підтримку проведення творчих письмових конкурсів. Така система може знайти застосування в процесах здобуття освіти та сфері бізнесу. Творчі письмові конкурси проводяться в усіх закладах освіти – це мовні олімпіади. Система підтримки таких заходів може значно спростити процес проведення самих конкурсів, позбавивши при цьому організаторів від зайвої паперової волокніти і дозволивши учасникам приймати участь у таких конкурсах не лише очно, а й на дистанційній основі.

Було описано предметне середовище, у якому функціонує система, та досліджено проблеми. Для вирішення проблем було визначено цілі дипломного проекту, поставлено задачі, виявлено призначення. Для проектування розглянуто існуючі аналоги серед представлених в мережі Інтернет і проведено порівняльний аналіз. Розроблено діаграму варіантів використання. Описано інформаційне забезпечення системи, розроблено структуру масивів даних, наведено реквізити вхідних та вихідних даних.

Побудовано алгоритм проведення тестового етапу конкурсу, якій порівняно з тривіальними і визначено переваги. До переваг відноситься значно зменшена кількість колізій, тобто кожен учасник отримає рейтинговий бал відмінний від усіх інших, та врахування балів, що передбачені за завдання організаторами одночасно з балами, які нараховуються фактично, виходячи з відповідей усіх учасників.

Розроблено проектні документи – схеми структурні компонентів, класів програмного забезпечення, послідовності. Наведено схему алгоритму роботи програмного забезпечення.

Було описано специфікацію функцій класів системи.

Надано розроблене керівництво користувача, в якому представлені функції, що забезпечені системою у відповідності до постановки задачі,

наведеної в першому розділі, та приведено рисунки відповідних графічних інтерфейсів та форм.

Проведено процес тестування програмного забезпечення і встановлено, що програмне забезпечення виконує усі заявлені функції у відповідності до передбачених результатів, що є бажаними з точки зору роботоздатності.

ПЕРЕЛІК ПОСИЛАНЬ

1. Стаття «Как и зачем писать Use Cases» [Електронний ресурс] // Режим доступу: https://dou.ua/lenta/articles/use-cases/
2. Творчий письмовий конкурс «Fansub Challenge» [Електронний ресурс] // Режим доступу: https://vk.com/fansubcollegium
3. Творчий письмовий конкурс «Письмо солдату» [Електронний ресурс] // Режим доступу: https://vsekonkursy.ru/festival-detskih-esse-pismo-soldatu.html
4. Бурлаков О.С., Мушеник І.М.: Оцінка якості тестових завдань діагностики знань студентів економічних спеціальностей засобами середовища дистанційного навчання moodle // Науково-виробничий журнал «Інноваційна економіка» 5-6’2016[63]
5. Exley K. Writing Good Exam Questions. A Self-study Workbook // London School of Hygiene & Tropical Medicine
6. Smith M.L. Put to the Test: The Effects of External Testing on Teachers / Mary Lee Smith // Educational Researcher. – June 1991. – P. 8-11
7. Офіційний сайт мови Java [Електронний ресурс] // Режим доступу: https://www.java.com/
8. Статя по роботі з JFrame [Електронний ресурс] // http://java-online.ru/swing-windows.xhtml
9. Портякин И.: Swing: Эффектные пользовательские интерфейсы // Лори, 2011 – 591 с.

Додаток А

Тексти програмного коду

Інформаційна система підтримки проведення творчих письмових конкурсів

(Найменування програми (документа))

DVD-R

(Вид носія даних)

20 арк, 23 Кб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2020 року

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;

public class AdminInterface extends JFrame implements Runnable  {

    Container container;

    public void setupContestCreator() {

        JLabel l2  = new JLabel("Create contest");
        l2.setBounds(700, 10, 120, 30);


        JLabel contestStage1Label = new JLabel("Stage 1 settings");
        contestStage1Label.setBounds(520, 50, 120, 30);
        container.add(contestStage1Label);

        JLabel hasFirstStageCheck = new JLabel("Qualifiers stage");
        hasFirstStageCheck.setBounds(520, 90, 120, 30);
        container.add(hasFirstStageCheck);

        JCheckBox qualBox = new JCheckBox();
        qualBox.setBounds(620,80,50,50);
        qualBox.setSelected(true);
        container.add(qualBox);

        ButtonGroup groupPoints = new ButtonGroup();

        JRadioButton pointButton1 = new JRadioButton("Single Point");
        pointButton1.setBounds(650,80,100,50);
        groupPoints.add(pointButton1);

        JRadioButton pointButton2 = new JRadioButton("Multi Point");
        pointButton2.setBounds(750,80,100,50);
        groupPoints.add(pointButton2);

        JRadioButton pointButton3 = new JRadioButton("Advanced Algo");
        pointButton3.setBounds(850,80,120,50);
        groupPoints.add(pointButton3);
        pointButton3.setSelected(true);

        container.add(pointButton1);
        container.add(pointButton2);
        container.add(pointButton3);


        JLabel contestStage2Labe1 = new JLabel("Stage 2 settings");
        contestStage2Labe1.setBounds(520, 135, 120, 30);
        container.add(contestStage2Labe1);

        JLabel stage2Type = new JLabel("Type");
        stage2Type.setBounds(520, 180, 120, 30);
        container.add(stage2Type);

        ButtonGroup groupTournType = new ButtonGroup();
```

```
JRadioButton groupButton1 = new JRadioButton("Tournament");
groupButton1.setBounds(650,175,100,50);
groupTournType.add(groupButton1);

JRadioButton groupButton2 = new JRadioButton("Round Robin");
groupButton2.setBounds(750,175,100,50);
groupTournType.add(groupButton2);

JRadioButton groupButton3 = new JRadioButton("Single Stage");
groupButton3.setBounds(850,175,120,50);
groupTournType.add(groupButton3);
groupButton1.setSelected(true);

JLabel assigned = new JLabel("Assigned Judges ids:        NONE");
assigned.setBounds(520, 225, 500, 30);
container.add(assigned);

JButton assignment = new JButton("Assign Judges");
assignment.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        JudgeAssigner judgeAssigner = new JudgeAssigner();
        judgeAssigner.setVisible(true);

        judgeAssigner.setJLable(assigned);

    }
});
assignment.setBounds(520, 270, 150, 30);
container.add(assignment);

container.add(groupButton1);
container.add(groupButton2);
container.add(groupButton3);

JButton createContestButton = new JButton("Create Contest");
createContestButton.setBounds(560, 350,350,80);
createContestButton.addActionListener(actionEvent -> {


    JLabel contestInfo = new JLabel("New contest");
    contestInfo.setBounds(20, 450, 200, 20);

    JButton manageButton = new JButton("Manage");
    JButton viewProgressButton = new JButton("Progress");
    manageButton.setBounds(210, 450, 100, 30);

    viewProgressButton.setBounds(330, 450, 100, 30);

    container.add(contestInfo);
    container.add(manageButton);
    container.add(viewProgressButton);
    container.repaint();


});
container.add(createContestButton);
container.add(l2);
```

```
    }

    public void setupContestList() {

        JLabel l1  = new JLabel("Active contests");
        l1.setBounds(200, 10, 120, 30);

        int xContestStart = 20, yContestStart = 50, yContestHeight= 40,
xContestLength = 200;
        int xJbutContestLeng=100, yjButContHeight=30;
        ArrayList<String> contests = new ArrayList<>();

        contests.add("Fansub Challenge");
        contests.add("ZNO");
        contests.add("Maths olympics");
        contests.add("Language olympics");
        contests.add("Taras Shevchenko's challenge");
        contests.add("United Writers");
        contests.add("My First Writing Expitience");
        contests.add("Letter to Daddy");
        contests.add("Kyyivstar Contest");
        contests.add("Home Sweet Home");

        ArrayList<JLabel> jLabelsContests = new ArrayList<>();
        ArrayList<JButton> jButtonsContests = new ArrayList<>();

        for (int i = 0; i < contests.size(); i++) {

            JLabel contestInfo = new JLabel(contests.get(i));
            contestInfo.setBounds(xContestStart, yContestStart + i*yContestHeight,
xContestLength, 20);
            jLabelsContests.add(contestInfo);

            JButton manageButton = new JButton("Manage");
            JButton viewProgressButton = new JButton("Progress");
            manageButton.setBounds(xContestLength+10, yContestStart +
i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(manageButton);
            viewProgressButton.setBounds(xContestLength+xJbutContestLeng+30,
yContestStart + i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(viewProgressButton);
            manageButton.addActionListener(actionEvent -> {
                ContestManager contestManager = new ContestManager();
                contestManager.setVisible(true);
            });

        }

        container.add(l1);

        for (var contest :
                jLabelsContests) {
            container.add(contest);
        }

        for (var but: jButtonsContests) {
            container.add(but);
        }
```

```
        }

        public AdminInterface() {
            super("Admin");
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            this.setBounds(448, 156, 1024, 768);
            this.setResizable(true);
            container = this.getContentPane();
            container.setLayout(null);
            setupContestCreator();
            setupContestList();

        }

        @Override
        public void run() {

        }
    }
```

```
import javax.swing.*;
import java.awt.*;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.InetAddress;
import java.net.Socket;

public class AuthInterface extends JFrame implements Runnable {

    Socket connection;
    private JTextField login = new JTextField("", 5);
    private JTextField password = new JTextField("", 5);
    private String loginString;
    private String passwordString;
    private JButton authentButton = new JButton("Login");
    private JButton registerButton = new JButton("Register");
    private ServerConnector connector;
    private ObjectOutputStream ostream;
    private ObjectInputStream istream;
    private int[] userType;

    private boolean connected = true;

    private String message;
    private String lastAnswer;

    public AuthInterface(int[] userType) {
        super("Authentification");
        this.userType = userType;
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setBounds(760, 400, 400, 200);
        this.setResizable(false);


        authentButton.addActionListener(actionEvent -> {
            loginString = login.getText().split("\n")[0].strip();
            passwordString = password.getText().split("\n")[0].strip();

            sendMessage("login\n" + loginString + "\n" + passwordString);

        });
        registerButton.addActionListener(actionEvent -> {
            loginString = login.getText().split("\n")[0].strip();
            passwordString = password.getText().split("\n")[0].strip();
            sendMessage("register\n" + loginString + "\n" + passwordString + "\n" +
"users");

        });


        Container container = this.getContentPane();
        container.setLayout(null);

        login.setBounds(40, 20, 300, 30);
        password.setBounds(40, 60, 300, 30);
        authentButton.setBounds(100, 110, 80, 30);
        registerButton.setBounds(200, 110, 80, 30);
        ;
```

```
        container.add(login);
        container.add(password);
        container.add(authentButton);
        container.add(registerButton);

    }

    @Override
    public void run() {

        try {

            while (connected) {
                connection = new Socket(InetAddress.getByName("127.0.0.1"), 4204);
                ostream = new ObjectOutputStream(connection.getOutputStream());
                istream = new ObjectInputStream(connection.getInputStream());

                String ans = (String) istream.readObject();
                if (!ans.equals("")) {
                    System.out.println("ans is "+ans);
                    rearrangeGroup(ans);

                    connected = false;
                    //connected = false;
                }

            }
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
        finally {
            try {
                connection.close();
                ostream.close();
                istream.close();
                setVisible(false);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

    }

    private void rearrangeGroup(String group) {
        if (group.equals("admin")) {
            userType[0] = 1;
        }
        if (group.equals("users")) {
            userType[0] = 2;
        }
        if (group.equals("judges")) {
            userType[0] = 3;
        }
    }

    private void paintUsers() {
        JLabel label = new JLabel("Your active contests");
        label.setBounds(10,10,100,15);
```

```
        }

        private void sendMessage(String message) {

            try {
                ostream.flush();
                ostream.writeObject(message);
                ostream.flush();
            } catch (IOException e) {
                e.printStackTrace();
            }


        }

    }
```

```
import javax.swing.*;
import java.awt.*;

public class ContestManager extends JFrame{

    public  ContestManager() {
        super("Contest Manager");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        this.setBounds(640, 300, 360, 400);
        this.setResizable(true);

        Container container = this.getContentPane();
        container.setLayout(null);

        ButtonGroup groupPoints = new ButtonGroup();

        JRadioButton pointButton1 = new JRadioButton("Single Point");
        pointButton1.setBounds(10,80,100,50);
        groupPoints.add(pointButton1);

        JRadioButton pointButton2 = new JRadioButton("Multi Point");
        pointButton2.setBounds(110,80,100,50);
        groupPoints.add(pointButton2);

        JRadioButton pointButton3 = new JRadioButton("Advanced Algo");
        pointButton3.setBounds(210,80,120,50);
        groupPoints.add(pointButton3);
        pointButton3.setSelected(true);

        container.add(pointButton1);
        container.add(pointButton2);
        container.add(pointButton3);

//
//        JLabel stage2Type = new JLabel("Type");
//        stage2Type.setBounds(20, 180, 120, 30);
//        container.add(stage2Type);

        JTextField newName = new JTextField("Enter new name");

        newName.setBounds(20,180,200,30);
        container.add(newName);

    }

}
```

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.util.ArrayList;

public class JudgeAssigner extends JFrame {

    Container container;

    private JLabel label;

    private int[] assignmentArray;

    private boolean IsAssigning = true;

    public boolean isAssigning() {
        return IsAssigning;
    }

    private void setupAvailableJudges() {
        JLabel availableJudjes = new JLabel("Available Judjes");
        availableJudjes.setBounds(100, 15, 120, 10);
        container.add(availableJudjes);

        JLabel assigned = new JLabel("Assigned");
        assigned.setBounds(150, 45, 120, 15);
        container.add(assigned);

//        JLabel notAssigned = new JLabel("Not Assigned");
//        notAssigned.setBounds(170, 45, 120, 15);
//        container.add(notAssigned);

        ArrayList<String> judges = new ArrayList<>();

        judges.add("Judge 1");
        judges.add("Judge 2");
        judges.add("Judge 3");
        judges.add("Judge 4");
        judges.add("Judge 5");
        judges.add("Judge 6");

        ArrayList<JLabel> jLabelsContests = new ArrayList<>();
        ArrayList<JCheckBox> jButtonsContests = new ArrayList<>();

        int xContentStart = 20, yContentStart = 80, yContentHeight = 40,
xContentLength = 100;
        int xJbutContentLeng = 60, yjButContHeight = 30;

        for (int i = 0; i < judges.size(); i++) {

            JLabel contestInfo = new JLabel(judges.get(i));
            contestInfo.setBounds(xContentStart, yContentStart + i * yContentHeight,
xContentLength, 20);
            jLabelsContests.add(contestInfo);

            JCheckBox manageButton = new JCheckBox();
```

```
            //JCheckBox viewProgressButton = new JCheckBox();
            manageButton.setBounds(xContentLength + 60, yContentStart + i *
yContentHeight, xJbutContentLeng, yjButContHeight);
            jButtonsContests.add(manageButton);

            manageButton.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent actionEvent) {

                    label.setText("Assigned judges ids: ");

                    for (int j = 0; j < jButtonsContests.size(); j++) {
                        if (jButtonsContests.get(j).isSelected()) {
                            label.setText(label.getText()+(j+1)+" ");
                        }

                    }

                }
            });
            //viewProgressButton.setBounds(xContentLength + xJbutContentLeng + 30,
yContentStart + i * yContentHeight, xJbutContentLeng, yjButContHeight);
            //jButtonsContests.add(viewProgressButton);
        }

        for (var contest :
                jLabelsContests) {
            container.add(contest);
        }

        for (var but : jButtonsContests) {
            container.add(but);
        }

//        JButton assign = new JButton("Assign");
//        assign.setBounds(90, 320, 100, 30);
//        assign.addActionListener(new ActionListener() {
//            @Override
//            public void actionPerformed(ActionEvent actionEvent) {
//                assignmentArray = new int[judges.size()];
//                for (int i = 0; i < judges.size(); i++) {
//                    assignmentArray[i] = jButtonsContests.get(i).isSelected() ? 1 :
0;
//                }
//            }
//        });
//        container.add(assign);

    }

    public void setJLable(JLabel label) {

        this.label = label;

    }

    public JudgeAssigner() {
        super("Judges");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
            this.setBounds(640, 300, 300, 400);
            this.setResizable(true);

            assignmentArray = new int[6];

            //this.addWindowListener();

            container = this.getContentPane();
            container.setLayout(null);

            setupAvailableJudges();

        }

    }
```

```
import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;

public class JudgeInterface extends JFrame{

    Container container;

    public void setupContestList() {

        JLabel l1  = new JLabel("Active contests");
        l1.setBounds(200, 10, 120, 30);

        int xContestStart = 20, yContestStart = 50, yContestHeight= 40,
xContestLength = 200;
        int xJbutContestLeng=100, yjButContHeight=30;
        ArrayList<String> contests = new ArrayList<>();

        contests.add("Fansub Challenge");
        contests.add("ZNO");
        contests.add("Maths olympics");
        contests.add("Language olympics");
        contests.add("Taras Shevchenko's challenge");
        contests.add("United Writers");
        contests.add("My First Writing Expitience");
        contests.add("Letter to Daddy");
        contests.add("Kyyivstar Contest");
        contests.add("Home Sweet Home");

        ArrayList<JLabel> jLabelsContests = new ArrayList<>();
        ArrayList<JButton> jButtonsContests = new ArrayList<>();

        for (int i = 0; i < contests.size(); i++) {

            JLabel contestInfo = new JLabel(contests.get(i));
            contestInfo.setBounds(xContestStart, yContestStart + i*yContestHeight,
xContestLength, 20);
            jLabelsContests.add(contestInfo);

            JButton manageButton = new JButton("Judge");
            JButton viewProgressButton = new JButton("Progress");
            manageButton.setBounds(xContestLength+10, yContestStart +
i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(manageButton);
            viewProgressButton.setBounds(xContestLength+xJbutContestLeng+30,
yContestStart + i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(viewProgressButton);

        }

        container.add(l1);

        for (var contest :
                jLabelsContests) {
            container.add(contest);
        }

        for (var but: jButtonsContests) {
            container.add(but);
```

```
            }

        }

        public JudgeInterface() {
            super("Judge");
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            this.setBounds(448, 156, 1024, 768);
            this.setResizable(true);
            container = this.getContentPane();
            container.setLayout(null);
            //setupContestCreator();
            setupContestList();
        }

    }
```

```
public class Main {

    public static void main(String[] args) throws InterruptedException {

        ServerConnector connector = new ServerConnector();

        int[] userType = new int[1];

        AuthInterface authInterface = new AuthInterface(userType);
        authInterface.setVisible(true);
        Thread runner = new Thread(authInterface);
        runner.start();
        runner.join();

        System.out.println(userType[0]);

        if (userType[0] == 1) {

            System.out.println("here");
            AdminInterface adminInterface = new AdminInterface();
            adminInterface.setVisible(true);
            Thread adminThread = new Thread(adminInterface);
            adminThread.start();
            adminThread.join();

        } else if (userType[0] == 2) {
            UserInterface ui = new UserInterface();
            ui.setVisible(true);
        }
        else {
            JudgeInterface ju = new JudgeInterface();
            ju.setVisible(true);

        }


//        JudgeAssigner assigner = new JudgeAssigner();
//        assigner.setVisible(true);




    }

}
```

```
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.InetAddress;
import java.net.Socket;

public class ServerConnector  {


    private String lastAnswer;
    private String message = "";

    public void sendMessage(String message) {
        this.message = message;
    }

    public String getLastAnswer() {
        return lastAnswer;
    }


}
```

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;

public class UserInterface extends JFrame{

    Container container;
    ArrayList<String> contests;
    JLabel assignedInfo;

    protected void setupContestList() {

        JLabel l1  = new JLabel("Active contests");
        l1.setBounds(200, 10, 120, 30);

        int xContestStart = 20, yContestStart = 50, yContestHeight= 40,
xContestLength = 200;
        int xJbutContestLeng=100, yjButContHeight=30;
        contests = new ArrayList<>();

        contests.add("Fansub Challenge");
        contests.add("ZNO");
        contests.add("Maths olympics");
        contests.add("Language olympics");
        contests.add("Taras Shevchenko's challenge");
        contests.add("United Writers");
        contests.add("My First Writing Expitience");
        contests.add("Letter to Daddy");
        contests.add("Kyyivstar Contest");
        contests.add("Home Sweet Home");

        ArrayList<JLabel> jLabelsContests = new ArrayList<>();
        ArrayList<JButton> jButtonsContests = new ArrayList<>();

        for (int i = 0; i < contests.size(); i++) {

            JLabel contestInfo = new JLabel(contests.get(i));
            contestInfo.setBounds(xContestStart, yContestStart + i*yContestHeight,
xContestLength, 20);
            jLabelsContests.add(contestInfo);

            JButton manageButton = new JButton("Sign");
            int finalI = i;
            manageButton.addActionListener(actionEvent -> {
                assignedInfo.setText(assignedInfo.getText().substring(0,
assignedInfo.getText().length()-7)+"<br>"+contests.get(finalI)+"</html>");
                container.repaint();
            });
            JButton viewProgressButton = new JButton("Progress");
            manageButton.setBounds(xContestLength+xJbutContestLeng+30, yContestStart
+ i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(manageButton);
            viewProgressButton.setBounds(xContestLength+10, yContestStart +
i*yContestHeight, xJbutContestLeng, yjButContHeight);
            jButtonsContests.add(viewProgressButton);

        }
```

```
            container.add(l1);

            for (var contest :
                    jLabelsContests) {
                container.add(contest);
            }

            for (var but: jButtonsContests) {
                container.add(but);
            }

        }

        protected void setupContestAssignment() {

            JLabel l2  = new JLabel("Current contests:");
            l2.setBounds(700, 10, 120, 30);
            container.add(l2);

        }

        public UserInterface() {
            super("User");
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            this.setBounds(448, 156, 1024, 768);
            this.setResizable(false);

            container = this.getContentPane();
            container.setLayout(null);
            assignedInfo = new JLabel("<html></html>");
            assignedInfo.setBounds(500,100,500,600);
            container.add(assignedInfo);
            setupContestAssignment();
            setupContestList();
        }

    }
```

```
import java.io.*;
import java.util.LinkedList;
import java.util.List;
import java.util.Scanner;

public class DatabaseInterractor {

    private List<User> users;

    public DatabaseInterractor() {
        users = new LinkedList<>();

        Scanner sc = null;
        try {
            sc = new Scanner(new File("users.mdf"));
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        assert sc != null;
        int userLength = sc.nextInt();
        sc.nextLine();

        for (int i = 0; i < userLength; i++) {
            User user = new User(sc.nextLine(), sc.nextLine(), sc.nextLine());
            users.add(user);
        }
        sc.close();

    }

    public String checkUser(User user) {

        System.out.println("checking user with data:\n" + user);

        String userInDB = "";

        for (var mUser : users) {
            if (mUser.getLogin().equals(user.getLogin()) &&
mUser.getPassword().equals(user.getPassword())) {
                userInDB = mUser.getGroup();
                break;
            }
        }
        System.out.println("user is: " + userInDB);
        return userInDB;

    }

    public void registerUser(User user) {
        if (checkUser(user).equals("")) {
            users.add(user);
            pushBD();
        }
    }

    private void pushBD() {

        try {
```

```
            BufferedWriter writer = new BufferedWriter(new FileWriter("users.mdf"));
            writer.write("" + users.size() + "\n");
            for (var user : users) {
                writer.write(user + "\n");
            }
            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

}
```

```
public class Main {
    public static void main(String[] args) {

        Server server = new Server();
        Thread serverRun = new Thread(server);
        serverRun.start();

    }
}
```

```
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;

public class Server implements Runnable {

    private Socket connection;
    private ServerSocket serverConnection;

    private ObjectOutputStream ostream;
    private ObjectInputStream istream;
    private DatabaseInterractor interractor;

    public Server() {
        interractor = new DatabaseInterractor();
    }
    @Override
    public void run() {

        try {
            serverConnection = new ServerSocket(4204, 10);
            while (true) {
                connection = serverConnection.accept();
                ostream = new ObjectOutputStream(connection.getOutputStream());
                istream = new ObjectInputStream(connection.getInputStream());

                ostream.flush();
                ostream.writeObject(runAction(istream.readObject()));
                ostream.flush();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
    public Object runAction(Object action) {

        System.out.println("Performing action\n"+action);
        String actionContent = (String) action;
        String actionHeader = actionContent.lines().findFirst().get();
        String[] actionInfo = actionContent.substring(actionContent.indexOf('\n') +
1).split("\n");

        if (actionHeader.equals("login")) {
            String group = interractor.checkUser(new User(actionInfo[0],
actionInfo[1], ""));
            if (group.equals("")) {
                System.out.println("Answering with USER NOT IN DATABASE");
                return "USER NOT IN DATABASE";
            } else {
                System.out.println("Answering with "+group);
                return group;
            }
        } else if (actionHeader.equals("register")) {
            interractor.registerUser(new User(actionInfo[0], actionInfo[1],
actionInfo[2]));
            System.out.println("Answering with users");
            return "users";
```

```
            }

            System.out.println("Answering with FAILED TO EXIT");
            return "FAILED TO EXIT";

        }
    }
```

```java
import java.util.Arrays;
import java.util.Objects;

public class User implements Comparable<User>{


    private String login;
    private String password;
    private String group;

    public String getGroup() {
        return group;
    }

    public String getLogin() {
        return login;
    }

    public String getPassword() {
        return password;
    }

    @Override
    public String toString() {
        return login+"\n"+password+"\n"+group;
    }

    public User(String login, String password, String group) {

        this.login = login.split("\n")[0].strip();
        this.password = password.split("\n")[0].strip();
        this.group = group.split("\n")[0].strip();

    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        User user = (User) o;
        return login.equals(user.login) &&
                password.equals(user.password);
    }

    @Override
    public int hashCode() {
        return Objects.hash(login, password);
    }

    @Override
    public int compareTo(User user) {
        if (user.login.equals(this.login) && user.password.equals(this.password)){
            return 0;
        }
        return -1;
    }
}
```

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО
Керівник проєкту
_____ *Тамара ТЄЛИШЕВА*
(підпис) (вл. ім’я, прізвище)
“13” квітня 2020 р.

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл. ім’я, прізвище)
“14” квітня 2020 р.

Інформаційна система підтримки проведення творчих письмових конкурсів

ТЕХНІЧНЕ ЗАВДАННЯ

Шифр *ДП 6314.01.000 ТЗ*

на 10 сторінках

Київ – 2020 року

ЗМІСТ

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Повне найменування системи та її умовне позначення

Повна назва системи: Інформаційна система проведення творчих письмових конкурсів.

Коротке найменування системи: «Система проведення конкурсів».

1.2 Найменування організації-замовника та організації-учасника робіт

Замовником є кафедра автоматизованих систем обробки інформації та управління Національного технічного університету України "Київський політехнічний інститут ім. Ігоря Сікорського" (далі за текстом — Замовник). Адреса замовника: м. Київ, п. Перемоги 37, 18 корпус ФІОТ.

Розробник сервісу — студент групи ІС-63 кафедри автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» Лопата Владислав Владиславович.

1.3 Перелік документів, на підставі яких створюється система

При розробці системи і створенні проектно-експлуатаційної документації Виконавець повинен керуватися вимогами наступних нормативних документів:

- ДСТУ 19.201-78. Технічне завдання. Вимоги до змісту і оформлення;
- ДСТУ 34.601-90. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення;
- ДСТУ 34.201-89. Інформаційні технології. Комплекс стандартів на автоматизовані системи. Види, комплексність і позначення документів при створенні автоматизованих систем.

1.4 Планові терміни початку і закінчення роботи зі створення системи

Плановий термін початку роботи над створенням інформаційної системи підтримки проведення творчих письмових конкурсів – 5 грудня 2019 року.

Плановий термін по закінченню роботи над створенням інформаційної системи підтримки проведення творчих письмових конкурсів – 30 травня 2020 року.

2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ

2.1 Призначення системи

Система призначена для організації конкурсів, що передбачають взаємодію учасників та журі.

2.2 Цілі створення системи

Ціль: покращити організацію, процеси проведення і оцінювання конкурсних творчих робіт за рахунок автоматизації видачі завдань та їх оцінювання.

Для досягнення цілі необхідно розв'язати наступні задачі:

- визначити функції системи;
- створити інформаційне забезпечення у вигляді масивів даних;
- розробити архітектуру системи;
- спроектувати інтерфейс;
- мінімізувати кількість можливих помилок шляхом тестування програмного продукту.

3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ

Для того, щоб користуватися системою, користувач повинен бути зареєстрованим та авторизованим. У залежності від типу користувача, необхідно створити різні типи акаунтів.

Об'єктами автоматизації є здача оцінювання робіт, побудова рейтингу та управління етапами конкурсу.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

Актор «організатор» може надавати завдання та визначати регламент проведення конкурсу. Актор «журі» може оцінити роз’язок завдання за умови, що актор «учасник» надішле розв’язок і актор «організатор» визначить систему балів за якою необхідно оцінити завдання. Актор «учасник» може отримати завдання, переглянути свою оцінку та надіслати розв’язок.

Для того, щоб кожен актор міг виконувати тільки функціонал, що передбачений саме для нього, необхідно на початку роботи програмного забезпечення проводити авторизацію, і тому усім акторам має бути надана можливість увійти в систему.

4.2 Вимоги до надійності

Система повинна функціонувати безвідмовно незважаючи на наявність ймовірних дефектів, які можуть проявлятися під час експлуатації. Виправлення таких дефектів повинно виконуватися на етапах розробки, бета-тестування та в процесі введення в дію і в межах промислової експлуатації.

Відмова програмного забезпечення сервісу не повинна призводити до руйнувань даних в інформаційних сховищах.

Будь-які аварійні ситуації повинні бути негайно задокументовані і надіслані розробникам задля усунення причин збою в роботі системи.

4.3 Вимоги до складу і параметрів технічних засобів

Мінімальні вимоги для технічного забезпечення клієнту:

- Оперативна пам'ять – 512 Мб;
- Місткість жорсткого диску – 1 Гб;
- Процесор – одноядерний, 1 ГГц;
- монітор, HD-графічна карта

Рекомендовані вимоги для технічного забезпечення клієнту:

- Оперативна пам'ять – 2 Гб;
- Місткість жорсткого диску – 1 Гб;
- Процесор – двоядерний, 1 ГГц;
- монітор, HD-графічна карта

Мінімальні вимоги для технічного забезпечення серверу:

- Оперативна пам'ять – 1 Гб;
- Місткість жорсткого диску – 5 Гб;
- Процесор – двоядерний, 1 ГГц;

Рекомендовані вимоги для технічного забезпечення серверу:

- Оперативна пам'ять – 4 Гб;
- Місткість жорсткого диску – 20 Гб;
- Процесор – восьмиядерний, 3 ГГц;

5 СТАДІЇ І ЕТАПИ РОЗРОБКИ

Основні етапи виконання робіт з розробки системи підтримки проведення творчих письмових конкурсів:

№ з/п	Назва етапу роботи	Термін виконання етапу	Результат виконання
1.	Підготовка технічного завдання на розробку програмного продукту	01.12.2019	
2.	Розробка сценарію роботи	01.01.2020	
3.	Технічне проектування – функціональність, модулі, задачі, цілі тощо	01.01.2020	
4.	Узгодження з керівником інтерфейсу користувача	08.01.2020	
5.	Розробка інформаційного забезпечення	20.01.2020	
6.	Розробка програмного забезпечення	07.02.2020	
7.	Налагодження програми	07.04.2020	
8.	Тестування програми	17.04.2020	
9.	Здача готового програмного продукту замовнику	22.05.2020	

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ

6.1 Види випробувань

Задля перевірки правильності роботи програмного продукту буде проведено функціональне тестування. В ході тестування буде виконано перевірку всіх функціональних характеристик застосунку. Також, система буде перевірена на відмовостійкість шляхом виконання некоректних дій користувачем. Окремими тестами буде перевірена безпека системи в цілому.

UNICHECK

Власник документу:
Попенко Володимир Дмитрович

ID перевірки:
1003947962

Дата перевірки:
11.06.2020 01:42:00 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
12.06.2020 01:40:00 EEST

ID користувача:
77149

Назва документу: **Lopata_is63**

ID файлу: **1003963111** Кількість сторінок: **37** Кількість слів: **5298** Кількість символів: **46368** Розмір файлу: **131.94 KB**

9.55% Схожість

Найбільша схожість: **2.94%** з джерело бібліотеки. ID файлу: **1003798192**

3.49% Схожість з Інтернет джерелами	18	Page 39
8.85% Текстові збіги по Бібліотеці акаунту	216	Page 39

0.08% Цитат

Цитати	1	Page 40

Вилучення переліку посилань вимкнено

0% Вилучень

Вилучений текст відсутній

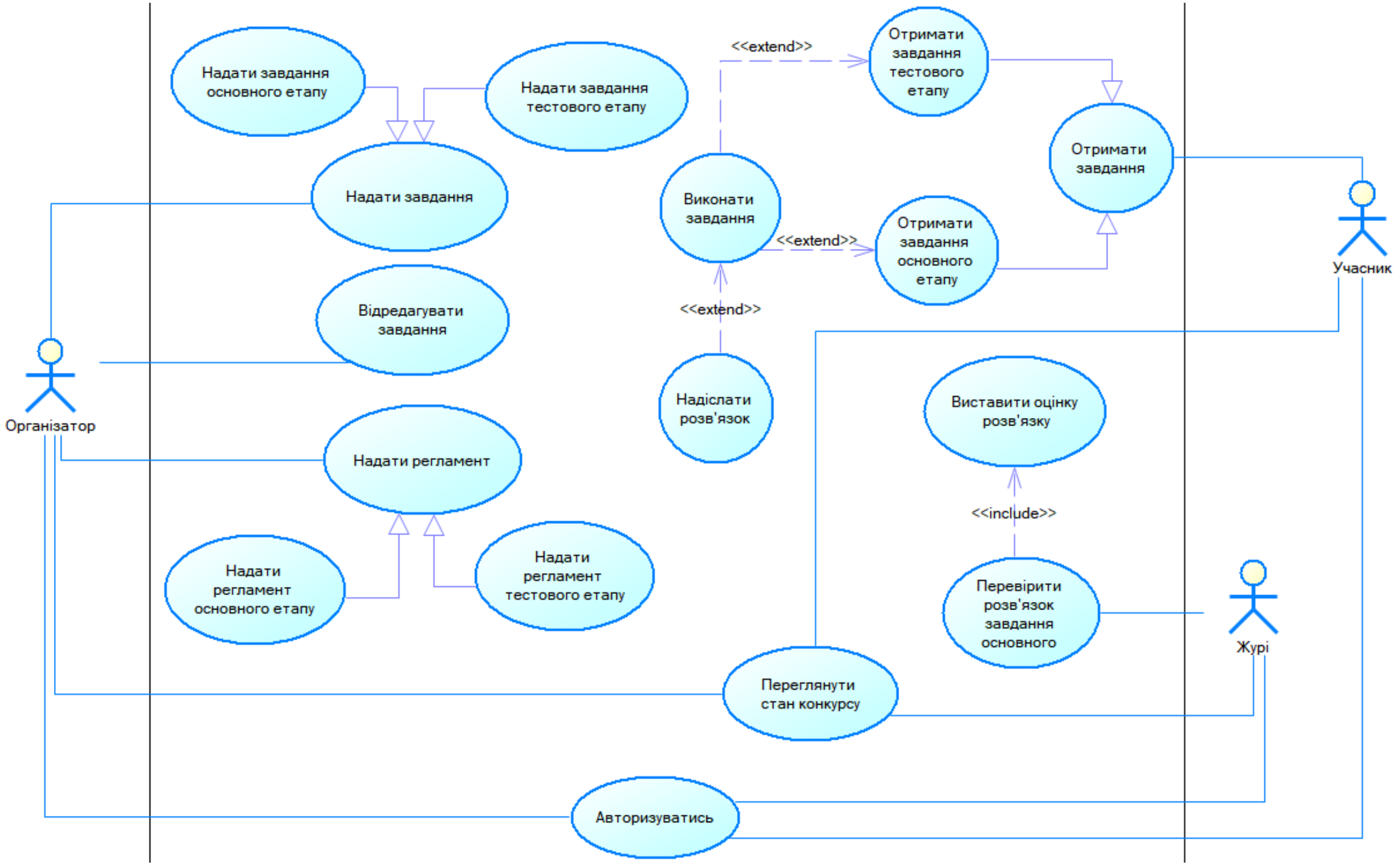
Підміна символів

Заміна символів	10

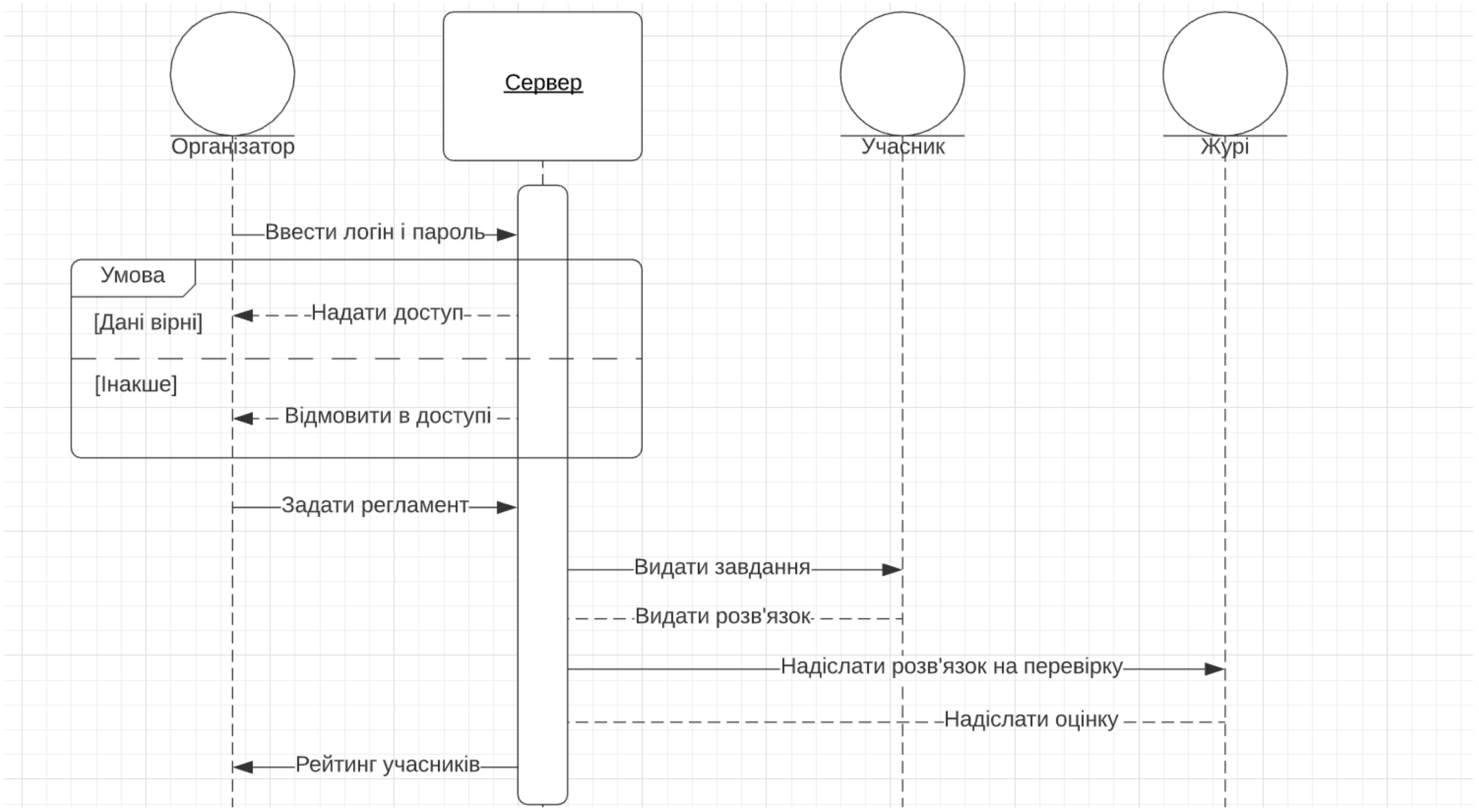
Графічний матеріал
до дипломного проєкту

на тему: «Інформаційна система підтримки проведення творчих письмових конкурсів»

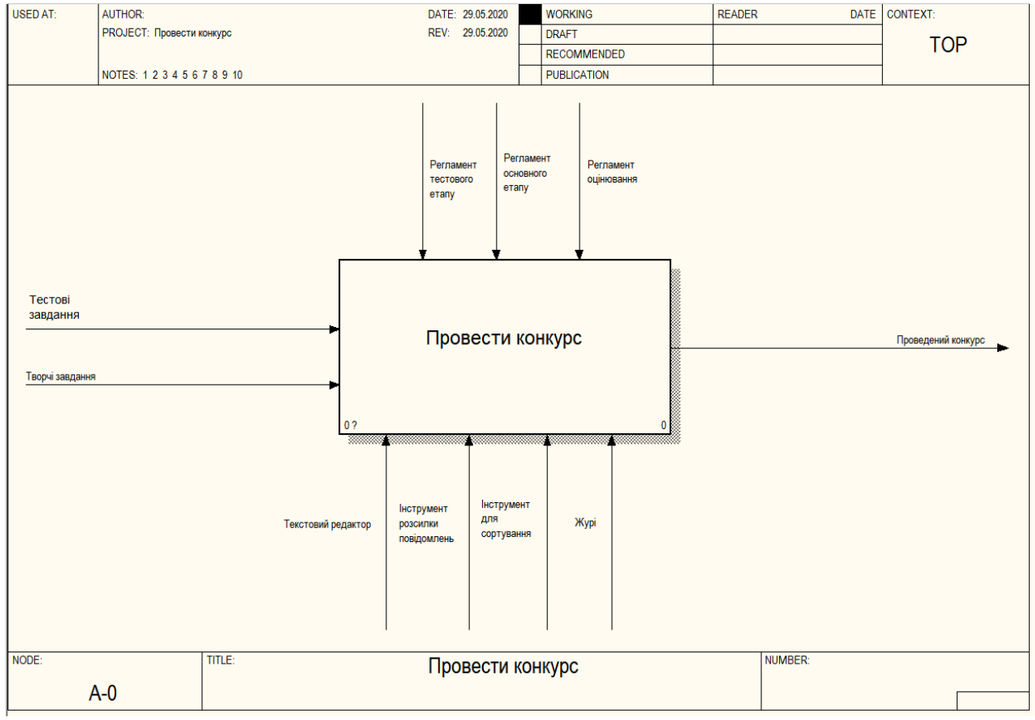
Київ – 2020 року

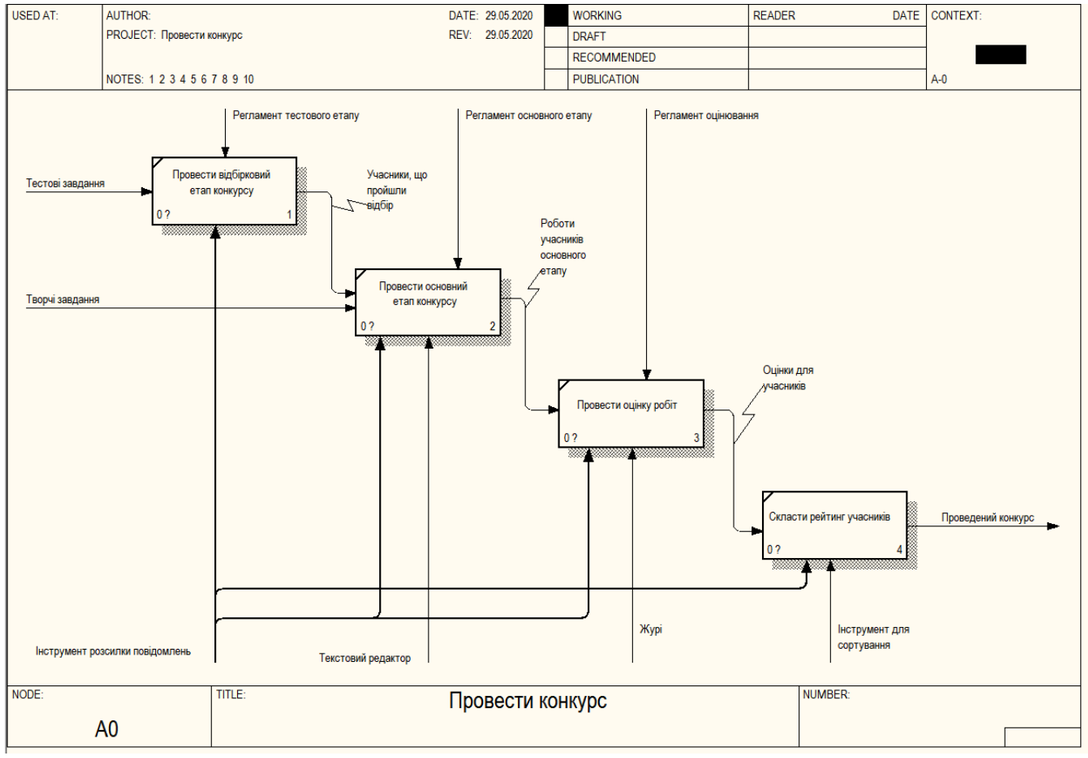


					ДП 6314.02.000 ССВ	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна варіантів використань	Літера / Маса / Масштаб
Розробив		Лопата В.В.				
Перевірив		Тєлишева Т.О.				Аркуш 1 / Аркушів 8
Т. кон.						
Н. кон.		Проскура С.Л.			Інформаційна система підтримки проведення творчих письмових конкурсів	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*
Затвердив		Тєлишева Т.О.				

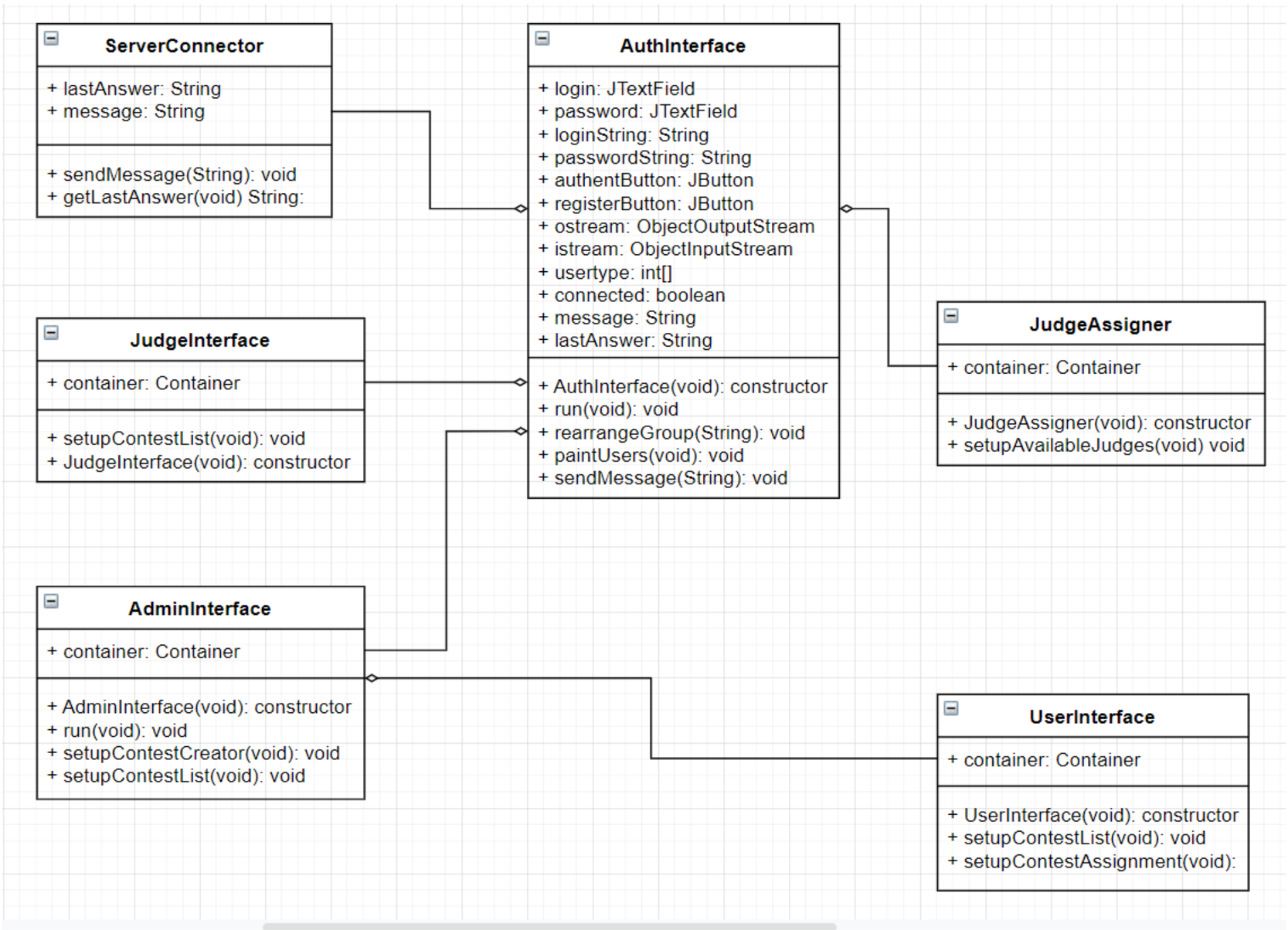


					ДП 6314.04.000 ССП	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна послідовності	Літера / Маса / Масштаб
Розробив		Лопата В.В.				
Перевірив		Тєлишева Т.О.				Аркуш 3 / Аркушів 8
Т. кон.						
Н. кон.		Проскура С.Л.			Інформаційна система підтримки проведення творчих письмових конкурсів	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*
Затвердив		Тєлишева Т.О.				



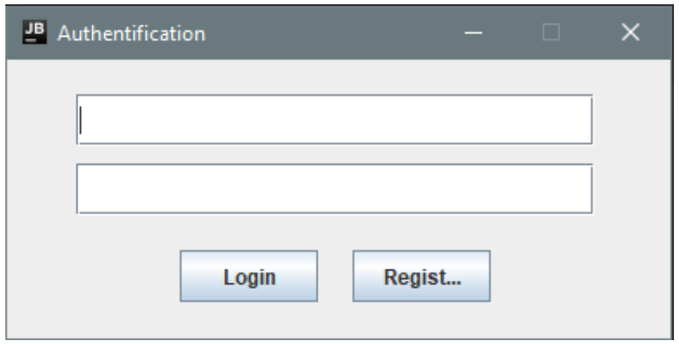


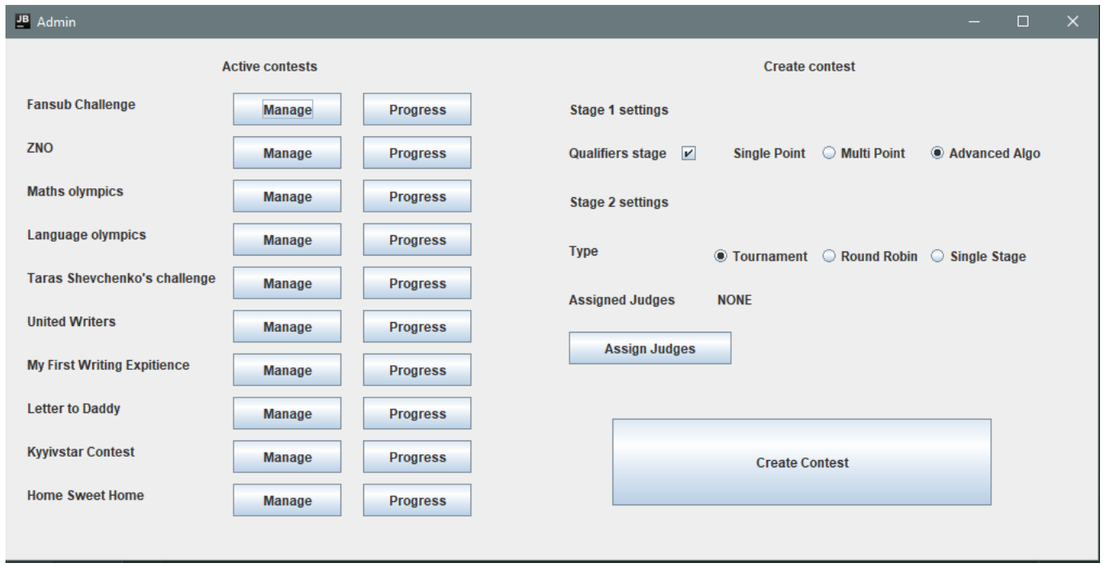
					ДП 6314.05.000		
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурно-функціональна процесу проведення конкурсу в нотаціях IDEF0	Літера	Маса / Масштаб
Розробив		Лопата В.В.					
Перевірив		Тєлишева Т.О.				Аркуш 4	Аркушів 8
Т. кон.							
Н. кон.		Проскура С.Л.			Інформаційна система підтримки проведення творчих письмових конкурсів	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*	
Затвердив		Тєлишева Т.О.					

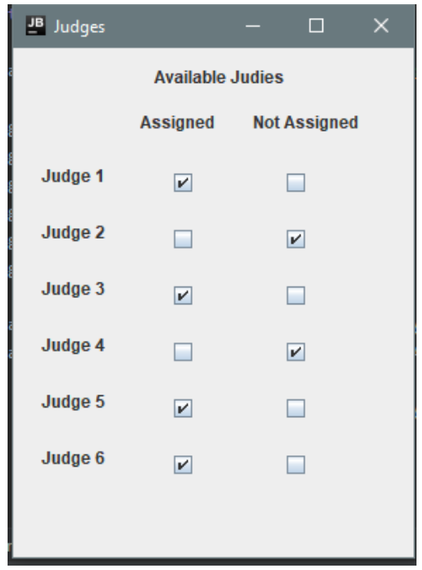


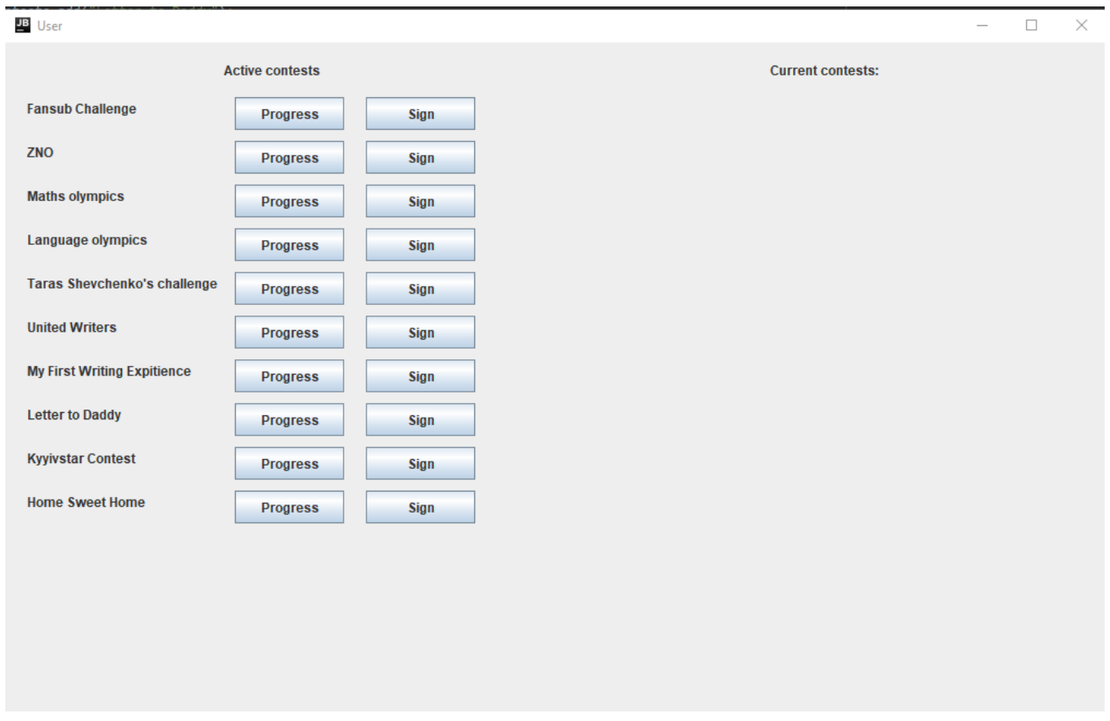
					ДП 6314.06.000 ССК	
					Схема структурна класів програмного забезпечення	Літера: — / Маса: — / Масштаб: —
Зм.	Арк.	№ документа	Підпис	Дата		Аркуш 5 / Аркушів 8
Розробив		Лопата В.В.				
Перевірив		Тєлишева Т.О.				
Т. кон.						
					Інформаційна система підтримки проведення творчих письмових конкурсів	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*
Н. кон.		Проскура С.Л.				
Затвердив		Тєлишева Т.О.				

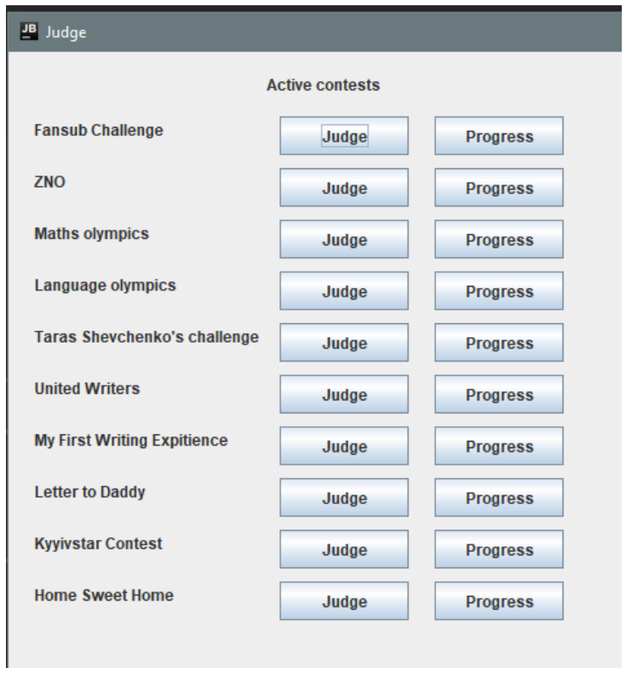
ДП 6314.09.000 КЕ



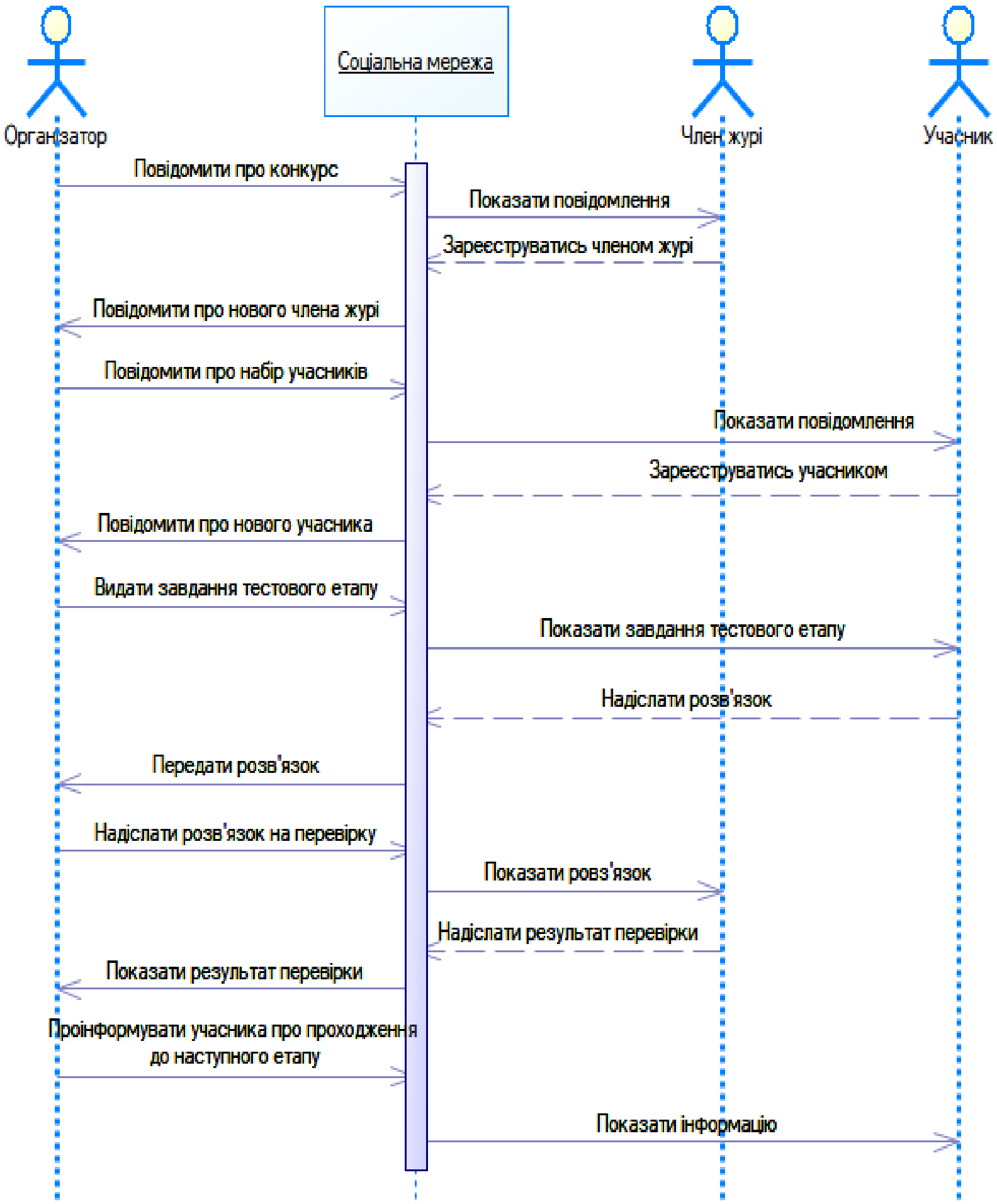




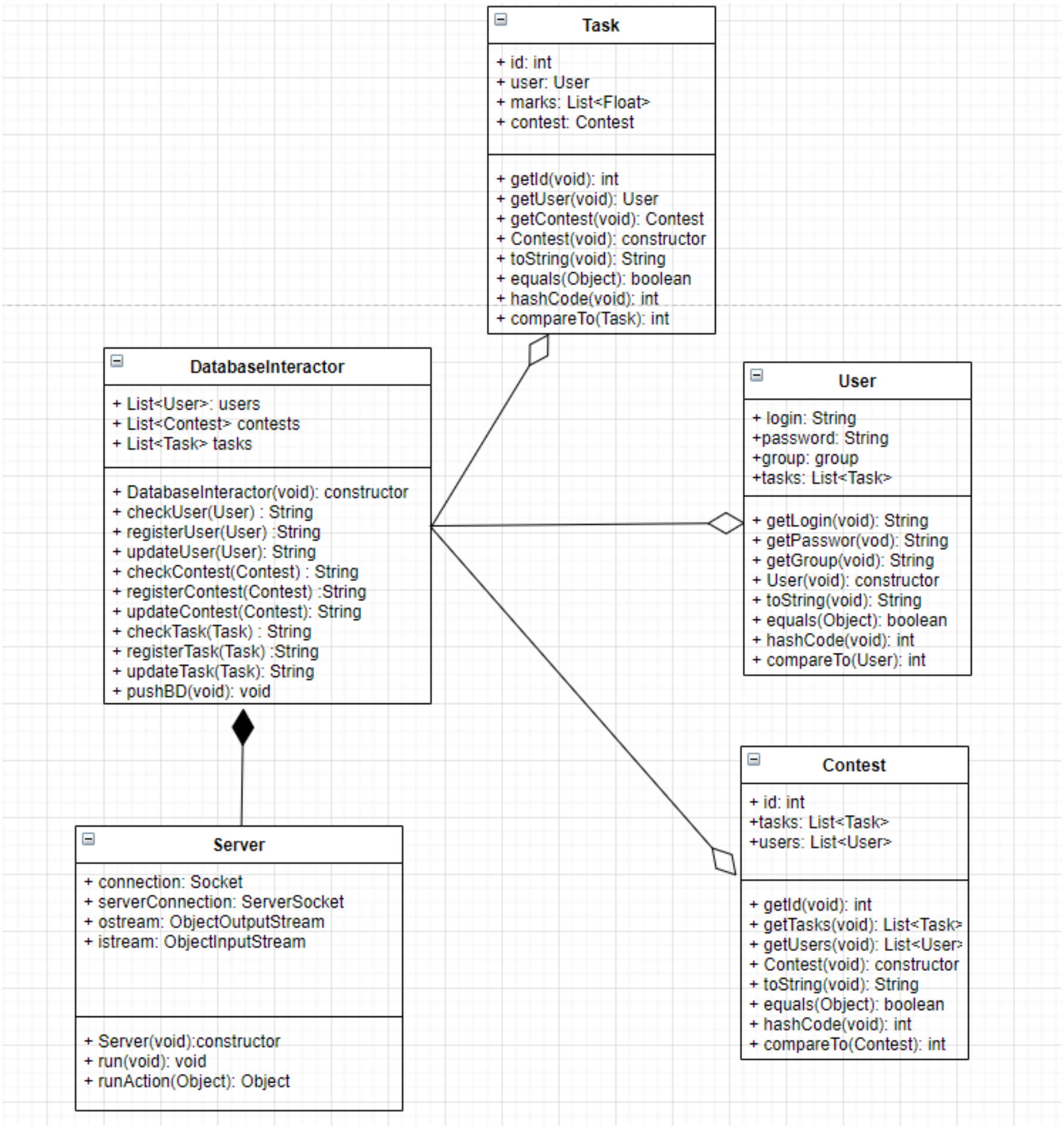




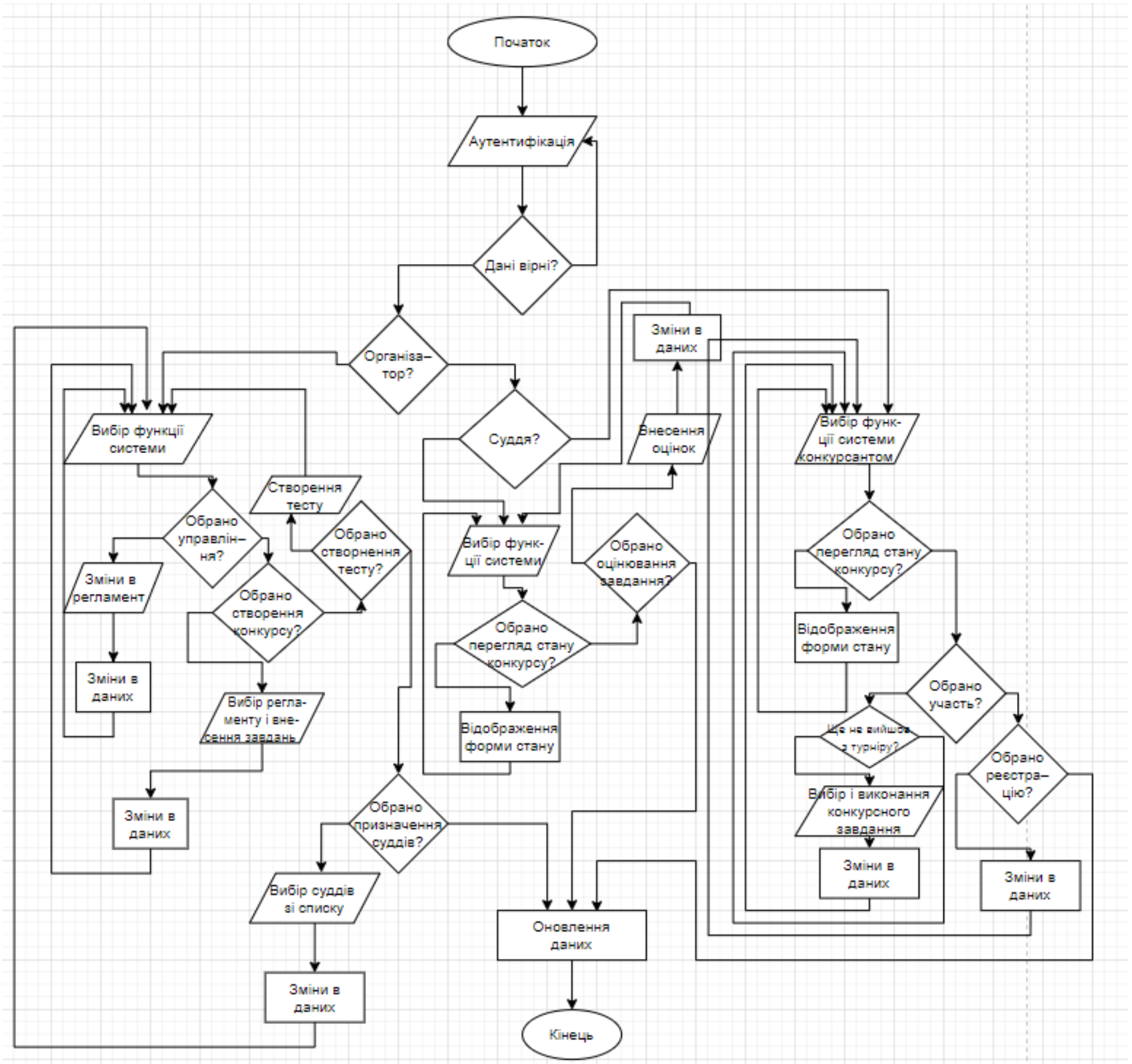
					ДП 6314.09.000 КЕ		
Зм.	Арк.	№ документа	Підпис	Дата	Креслення вигляду екранних форм	Літера	Маса / Масштаб
Розробив		Лопата В.В.					
Перевірив		Тєлишева Т.О.				Аркуш 5	Аркушів 8
Т. кон.							
Н. кон.		Проскура С.Л.			Інформаційна система підтримки проведення творчих письмових конкурсів	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*	
Затвердив		Тєлишева Т.О.					



					ДП 6314.03.000 ССП	
					Схема структурна послідовності	Лит. / Маса / Масштаб
Зм.	Арк.	№ докум.	Підп.	Дата		Аркуш 2 / Аркушів 8
Розроб.		Лопата В.В.			Інформаційна система підтримки проведення творчих письмових конкурсів	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-41
Перев.		Тєлишева Т.О.				
Т. Кон.						
Н. Кон.		Проскура С.Л.				
Затв.		Тєлишева Т.О.				



					ДП 6314.07.000 ССК	
					Схема структурна класів програмного забезпечення	Лит. / Маса / Масштаб
Зм.	Арк.	№ докум.	Підп.	Дата		
Розроб.		Лопата В.В.				Аркуш 6 / Аркушів 8
Перев.		Тєлишева Т.О.			Інформаційна система підтримки проведення творчих письмових конкурсів	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-41
Т. Кон.						
Н. Кон.		Проскура С.Л.				
Затв.		Тєлишева Т.О.				



					ДП 6314.08.000 СА	
Зм.	Арк.	№ докум.	Підп.	Дата		
Розроб.		Лопата В.В.			Схема алгоритму	Лит. / Маса / Масштаб
Перев.		Тєлишева Т.О.				Аркуш 6 / Аркушів 8
Т. Кон.					Інформаційна система підтримки проведення творчих письмових конкурсів	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-41
Н. Кон.		Проскура С.Л.				
Затв.		Тєлишева Т.О.				